# PRAGMA

A publication of Semaphore Corporation

Issue Number 4 May, 1983

## IN THIS ISSUE

## DEPARTMENTS

# Why wait?

Your PC can now be a ''Personal Minicomputer'' with the **Revelation Operating System** from Cosmos.

**Revelation** is a Relational Data Base Management System for the Personal Computer that provides minicomputer power at microcomputer prices. You can now have a small, portable computer that will serve as a working tool on the job, at home or on the road for less than $5,000.

**Revelation** is PICKTM compatible and operates on the IBMTM Personal Computer, as well as the Columbia Multi-PersonalTM Computer and the COMPAQTM.

As a Relational Data Base Management system, **Revelation** operates on top of MS-DOSTM, allowing you to switch back and forth between either operating environment and utilize hundreds of existing programs.

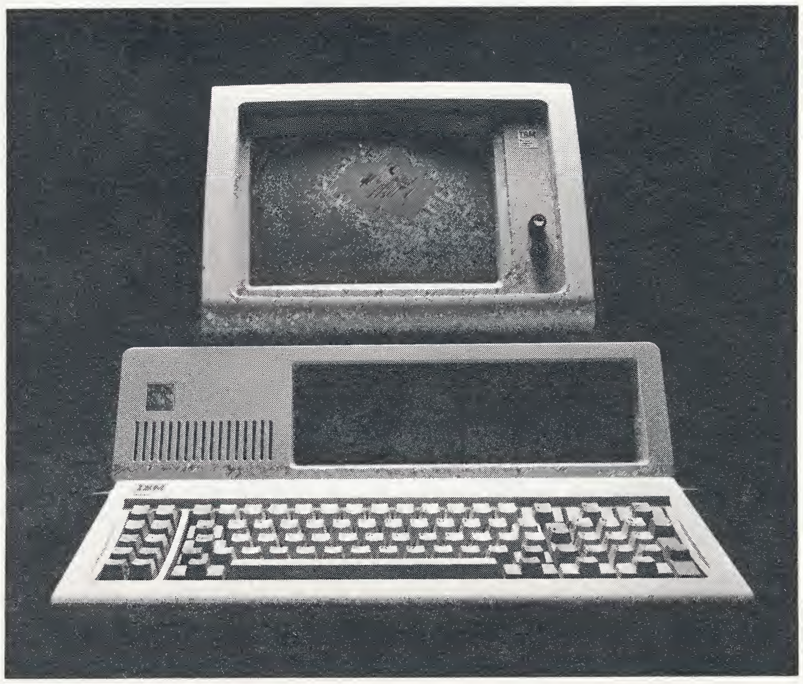
Relational Data Base Management and **Revelation** provide easy access to your data on floppy or hard disk. You use meaningful field names to refer to the data you desire. You can also define relationships that exist between two or more fields.

**Revelation** employs the Intel 8087 math chip on the PC for powerful number crunching capacity, floating point arithmetic and hardware mathematical functions.

Our application documentation and generation tools make **Revelation** ideal for software development and maintenance. And **Revelation's** advanced technology includes source code generation so you can do any job that needs to be done.

Communications problems between minicomputers and microcomputers are solved as **Revelation** allows your machine to communicate with compatible minicomputers.

Now you can afford the advantages of distributed processing.


IBM Personal Computer

Collect data at a remote site during the day and ship it to your minicomputer when you want. You can also use your PC as a dumb terminal to initiate jobs on your local or remote minicomputer.

System Memory hassles are eliminated. Just add one of the many available 512K RAM boards to your microcomputer and you have the memory you need to do your work quickly and efficiently with **Revelation.**

Now you can have minicomputer capabilities at microcomputer prices.

From Cosmos.

**Sample configuration**

**Sample IBM System**

| | |
|---|---|
| • System unit. 64K RAM keyboard. 160K disk drive and disk drive adapter | |
| **System price** | $2.205 |
| • 2nd 160KB disk drive | $450 |
| • Monochrome display | $345 |
| • Monochrome display and printer adapter | $335 |
| • 8087 math chip | $220 |
| • 256K RAM with serial interface | $349 |
| • Revelation software | $950 |
| • MS-DOS | $40 |
| **Total** | **$4.894** |

**Minimum configuration**

- 320K RAM
- 8087 math chip
- MS-DOS or PC-DOS
- One single-sided floppy drive

COSMOS

**P.O. Box AH**
**123 Ferntree Drive W.**
**Morton, WA 98356**
**(206) 496-5974**
**24-hour answering service: (206) 226-9362**

MS/DOS TM of Microsoft Corp Intel TM of Intel Corp ADDS Mentor TM of Applied Digital Data Systems. PICK Operating System TM of PICK SYSTEMS COMPAQ Portable Computer TM of COMPAQ Computer Corp Columbia Multi-Personal Computer is registered TM of Columbia Data Products Inc

# Survey Says...

Pragma has been feverishly compiling all kinds of data from four different surveys for presentation in this issue. Results from our reader survey have been arriving, and we have found the responses to be extremely valuable. (The questionnaire appeared in February's Pragma.) The only regular departments consistently rated high by readers appear to be Utilities, Command Files and Queries. All other departments were rated high as often as they were rated low, usually depending on the job description of the reader: managers preferred our non-technical departments and articles, while programming and operations personnel preferred to see technical content.

One exception was the User Profile, which was rated lower by DP managers more often than we would have expected. Since we felt that the interviews always contain a wealth of interesting information for the reader, we called a few of our local subscribers who are DP managers and tried to get a better understanding of why a particular user profile might not be considered interesting. The answer was often something like "The interview was a waste of time." We would counter with something similar to "But don't you think the hint about the bug that was revealed on page x of the interview was valuable?", and the response was often "Well, I actually didn't read that far." Soon the trend became apparent — DP managers (who often felt rushed and pressed for time by their jobs) would frequently read a paragraph or two, and if the story didn't immediately appear interesting, skip the rest of the interview!

Well, it's hard to please all of the people all of the time, especially when features are rated high as often as they are rated low, so we've decided the best solution is to try and vary the length of the various departments from issue to issue, so that sooner or later every reader gets equal time.

Most of the readers' responses did not offer any ideas concerning what new features to add to Pragma. Predictably, readers usually asked for more of what they liked and less of what they didn't like. However, a number of the forms did request more coverage of local user groups, so a new department doing just that is debuting in this issue. One reader asked us to "get a little more practical." Hmmm. We thought we already were so pragmatic that Pragma was an appropriate name, but apparently we have a ways to go!

Interestingly, almost every questionnaire that was returned came from a Microdata™ site. We happen to know that Pragma is reaching subscribers who represent every type of Pick system hardware, so does that mean only Microdata sites are interested in giving us feedback? More likely, the apparently lopsided return is simply because Microdata still controls the vast majority of Pick sites. To see what we mean, check the number of installations claimed by each vendor in the survey of hardware appearing elsewhere in this issue. Those numbers back up the idea that receiving responses only from Microdata sites is apparently not so unusual, statistically speaking.

In that survey, which shows the minimum and maximum hardware configurations that are available, the vendors appear capable of supporting quite a large range of machine capacities. But in our third survey (originally mentioned alongside our February questionnaire), we asked to hear from users who felt they were using the biggest or smallest hardware configuration that supports a Pick or Pick look-alike operating system. Roger Harpel of Cosmos wrote to claim the title for fewest ports (two serial and one parallel) and smallest disk (320 KB of floppies) with their Revelation product for the IBM personal computer. (Yes, they're a vendor, but Roger's letter assured us that the unit described is "for an operational configuration that we use daily in our business.") Honors for smallest amount of memory go to the San Francisco Police, with Ray Wong reporting only 32 KB of memory on a Microdata at that site. The high end record is currently being held by Computerized Automotive Management Systems, also in San Francisco, with Louis Beltjens claiming 48 ports, 256 MB of disk and 1 MB of memory on a Sequel™. Aren't there any bigger systems out there?

Our fourth survey, also found in this Pragma, is a comparison of BASICs that are supported by a selection of vendors. We think it shines some light on just how "compatible" Pick-style systems really are. Or should we say aren't?

•••

*Combining simplicity of use with data base management, Pick's operating system is the essence of processing information for management control with the lowest possible overhead.* That was the $250 winning entry by Pragma subscriber Bruce Slawinski of Anaheim CA in the "Why I like My Pick Operating System" contest as advertised in the February Pragma. Second place $100 winners were Fran Jacobson of Culver City CA and T.J. Leifield of Boulder CO. Third place winners of Pocket Guides were Janice Bartlett, also of Boulder, Gail Bellon of Rolling Meadows IL, Jay Bonham of Irvine CA, Anna Carsen, Steve Mattson and Tom Welch, all of Clearwater FL, Peter Coy of Toronto ON, Cheryl Guminik of Detroit MI, Shirley Harvey of Los Angeles CA, Phil Huntingdon of Anchorage AK, Joseph Sirota of Culver City CA and Wesley Woodland of Murray UT.

•••

Speaking of winners, Karen Anderson of Tolbey Equipment in Glendale AZ became name #6000 on our mailing list in March. In appreciation of her part in helping us set that milestone, and so Karen could join us in celebrating, we sent her a complimentary box of See's chocolates. Thank you, Karen!

*—The Editors*

**Computer-Generated Index Now Ready**

A complete, machine-generated master index of all articles in the first four issues of Pragma has been generated and will appear in Pragma #5 as a reference tool for its readers. Articles and features are extensively cross-referenced by words found in each article's title, by any personal names that appear, by additional miscellaneous keywords for various subject areas, and by department. Also, all advertisements are indexed by vendor, product name and product type.

# PRAGMA

**Issue Number 4** **May, 1983**

**Pragma** is published at least four times a year by

**Semaphore Corporation**
**207 Granada Drive**
**Aptos, California 95003**

## SUBSCRIPTIONS

Subscriptions are $25 per issue, $100 per year (four issues) in the USA, $38 per issue to other countries by airmail. All payments must be in US dollars drawn on a US bank.

Paid subscribers who refer new subscribers will receive a free subscription extension of one issue for each referral. Such referrals must include an appropriate subscriber number as explained on subscription order forms.

Address all subscription correspondence to the **Pragma** Circulation Manager, **Semaphore Corporation.** When writing, enclose your issue's mailing address label or the numbers from the upper right corner of your label.

Address changes should be sent at least four weeks in advance. Include old and new addresses along with your issue's mailing address label.

## SUBMITTALS

All correspondence and material received will be considered for publication. Except for correspondence used in the Letters Department, authors are paid up to $200 per full published page for submitted material used in **Pragma.** Actual payment amounts are decided by the Editors and vary with the amount of required editing and rework and with the length of each submittal. Authors are also granted free subscription extensions of one issue for submittals of at least one full published page. Address all submittals and correspondence to the **Pragma** Editors, **Semaphore Corporation.** All letters to the Editors are welcome and as many as possible will be published in the Letters Department.

All submittals must be typed on white paper and double spaced. The first page of any submittal and any accompanying material must include the author's name, address, telephone number and the date. All pages must be numbered.

Hand-typed program listings and other simulated printouts will not be accepted. Authors should submit actual computer-generated output. Print all listings with a fresh black ribbon on continuous white paper. Do not print on perforations. Do not include page numbers or headings on listings in order to simplify reduction and layout. Accompanying documentation should refer to listings by content, line number or symbolic labels, not by page number.

Drawings, schematics and other illustrations must be in black ink on white paper and drawn in a large scale to allow significant reduction. Photographs must be black and white glossies.

Manuscripts are submitted at the author's risk. Unused manuscripts will be returned if a stamped, self-addressed envelope is included. Requests to review galleys must accompany the manuscript when it is first submitted. The Editors reserve the right to edit all submittals.

## ADVERTISING

Send all advertising correspondence, requests for advertising rates, and advertising copy to the **Pragma** Advertising Manager, **Semaphore Corporation.** Advertisers sending press releases are requested to telephone the Advertising Manager for an interview at 408-688-9200 within two weeks after submitting the material.

## READER SERVICES

Is there a service **Semaphore Corporation** can provide for you? Address all inquiries to **Pragma** Reader Service, **Semaphore Corporation,** or telephone 408-688-9200.

**BACK BY POPULAR DEMAND**

# The solution to those confusing reference manuals!

**FROM INFORMATION RESEARCH OF NEW YORK**

Need some help understanding your Microdata™ computer? Then you need Osrow's Reality® Technical Journal, from Information Research of New York. The Journal's pages are packed with detailed explanations, discussions and examples showing how to use and operate Microdata computers. Written in an easy to understand tutorial format, the Journal has been read and applied by Microdata users everywhere. Whether you are a novice or an expert, the Journal is the perfect complement to confusing and opaque reference manuals.

**SECTION I**
Security: Protecting an Account, System Level Privileges, Monitoring System Usage with the ACC File, Protecting Program Files and Source Code, Security Verbs, Protecting a File, Locking Up Your System, Protecting an Attribute, Protecting a Terminal, Reserving a Terminal, Program Protection with Table Lookup • Learning the ENGLISH® Language, Dictionary Synonyms, Sorting Problems, The F Stack • System Efficiency: File Reallocations, The Restore Method, The Tape Method, The Copy Method • Debugging Procs • Converting Dates • Editor: Commands, Dealing with Lines, Multiple Prestore Commands • Reading an Item and Extracting an Attribute, Program and System Locks, The Program Debugger.

**SECTION II**
Creating User Accounts: Account Creation Schemes, Defining the New Account, Workspace Assignment, Q-Pointers, Proc Pointers • Cleaning Up Accounts and Removing: Dangerous Verbs/Procs, the Editor Ability, the ENGLISH Ability, the Bisync, SCREENPRO™ and Tape Verbs, the Spooler and Programming Verbs • Creating a Low Priority Account Creation Proc, Synonym Accounts • A Proc Primer • Security: Backing Up on Tape and Paper, File-Saves, Account-Saves, Dumping Files, Selective Restores • Dealing with Data: the Different Ways Data is Represented • Macros • ENGLISH: Creating Labels, F-Stacks • The RUNOFF™ Fill, Nofill and Justify Commands • Editor: Prestore Looping • Replacing Attributes and Writing Items, The READV and WRITEV Commands, The MATREAD and MATWRITE Commands, The Program Debugger.

**SECTION III**
Menu Construction • System Utilities: The Spooler, The SP-STATUS Verb, Assigning the Printer, Creating Hold Files, Printing Hold Files, A Proc to Print Hold Files, A Proc to Delete Hold Files • Proc: Transfers, Pointers, Jobstream Procs, A Proc Primer, The O Command, Screen Formatting, The X Command, The D Command • ENGLISH: The SELECT and SSELECT Verbs • List Management: Saving Items in a List.

**SECTION IV**
A Complete Program to Find the Report Length of an ENGLISH Proc: Introduction, Source, Listing, Narrative • Security: Controlling Electricity, Fire Protection, Protection Against Water Damage, Controlling Static Electricity • ENGLISH: Default Dictionary Attributes for Automatic Listings • A Proc Primer • System Utilities: The Spooler, The SP-KILL Verb • Editor: F, FS, FI, FD and EX Commands • Efficiency: Scheduling Jobs at Night, The TIMESLICE Verb, The POVF Verb.

Want to learn more about using your Microdata computer? Do you need simple explanations telling how to make your computer sit up and beg? Order Osrow's Journal today! Available exclusively by subscription for $125 since 1981, **you can use the coupon to order your Journal for only $49!**

Name ______________________

Company ______________________

Address ______________________

City ______________ State/Province ______________

Country ______________ ZIP/Mail Code ______________

Telephone ______________________

Please send me ______ Journal(s) at $49 each. The Journal contains all four sections listed. SUBTOTAL ________

Add $4 per Journal for North America shipping. Outside of North America, add $7.50 per Journal. SHIPPING ________

New York residents, please add appropriate sales tax. TAX ________

Make checks payable to Information Research of New York. Mail your check and this form to: Osrow's Journal, 207 Granada Drive, Aptos, CA 95003. TOTAL ________

# SYSMAP A Cross-Reference System Part 4: Dicts and Procs

**This fourth article in a series on the use of cross-references presents all dictionary word definitions for SYSMAP files, and all procs for generating SYSMAP reports.**

The previous three installments presented all SYSMAP files and all the input programs necessary to manually fill the files with cross-reference data. The listings on this page are all of the necessary dictionary word definitions for each of the six SYSMAP files, so that procs can be built to generate various cross-reference reports that output the files in a meaningful and useful format.

The menu proc on page 9 is a master proc that can be used to invoke all of the input programs and all of the SYSMAP procs that generate reports similar to the samples shown at the bottom of that page. The sample reports happen to show some of the data that documents SYSMAP itself. For example, sample report #18 is showing that the third XF file attribute is read and written by the GET.XF program.

Fourteen different reports can be generated by the master proc, which assumes all SYSMAP procs are kept in a library file called SPL. The individual report generating procs are all shown on page 10.

The next and final SYSMAP installment will discuss approaches to making SYSMAP a fully automated system that does not depend on manual input for generating the data stored in the cross-reference files.

```
FF............. D/CODE A/AMC S/NAME............. V/CORR.... V/TYP V/MAX

ATR             S      0     Attribute           G1*1       L     15
FILE.NAME       S      0     File                G*1        L     10
AMC             S      1     AMC                            R     3
AVS             S      2     *                              L     1
DESC            S      3     Description                    T     45

XB............. D/CODE A/AMC S/NAME............. V/CORR.... V/TYP V/MAX

PROG            S      0     Program                        L     20
PROCS           S      1     Procs                          L     40
                             using
                             program

XD............. D/CODE A/AMC S/NAME............. V/CORR.... V/TYP V/MAX

FILE.NAME       S      0     DICT For            G*1        L     10
WORD            S      0     Word                G1*1       L     20
PROCS           S      1     Procs                          L     25
                             using
                             word
WORDS           S      2     Words                          L     25
                             using
                             word

XF............. D/CODE A/AMC S/NAME............. V/CORR.... V/TYP V/MAX

AMC             S      0     AMC                 TFF;X;;1   R     3
ATR             S      0     Attribute           G1*1       L     15
FILE.NAME       S      0     File                G*1        L     10
RPROG           S      1     Progs                          L     20
                             that
                             read (and use)
UPROG           S      2     Progs                          L     20
                             that
                             update
WORDS           S      3     Dictionary words               L     25
                             that
                             read

XP............. D/CODE A/AMC S/NAME............. V/CORR.... V/TYP V/MAX

PROC            S      0     Proc                           L     25
PROCS           S      1     Procs                          L     30
                             using
                             proc

XT............. D/CODE A/AMC S/NAME............. V/CORR.... V/TYP V/MAX

AMC             S      0     AMC                 TFF;X;;1   R     3
ATR             S      0     Attribute           G1*1       L     15
AVS             S      0     *                   TFF;X;;2   L     1
FILE.NAME       S      0     File                G*1        L     10
PROGS           S      1     Progs                          L     10
                             that
                             translate
PROCS           S      2     Procs                          L     25
                             that
                             read/write/translate
SCREENS         S      3     Screens                        L     15
                             that verify or
                             translate
FILES           S      4     Files               G*1        L     10
                             that
                             translate
WORDS           S      4     Words               G1*1       L     20
                             that
                             translate
```

**SYSMAP Dictionary Definitions**

## NOW anyone can create data processing applications with AIDS™!!!

AIDS™ is the system that makes it easy for users who have no data processing experience to build their own data base and retrieve information from it. This is done without the user having to write a single line of program code.

- Data files are created and dictionaries are defined by the user being prompted through a few easy steps. If the user has questions about how to use AIDS™, on-line help messages and a user manual written in non-technical language guide the user through the necessary steps.
- The data entry program uses the data fields defined in the dictionary to prompt the user to enter data. This program is also used to recall, review and if necessary, change or delete previous entries.
- Reports are created by a process that assists the user in building English-like data base inquiry sentences. It allows the user to save these sentences, recall and change them, and to run reports at any time.

The price of AIDS™ is only $1195.

AIDS™ is available for all computers that use the Pick or a Pick-type operating system. And you can order AIDS™ with a 30 day unconditional money back offer. Just complete and mail the software license agreement shown below with your check for $1195 (Colorado residents add 3½% sales tax).

(303) 773-2826

**thesoftwaregroup** INC.

P.O. Box 3082, Englewood, CO 80155-3082

Licensee: Company Name ______________________

Address ______________________

City ______________ State ________ Zip ________

Under this agreement, The Software Group, Inc. grants a licensee a perpetual, non-exclusive, non-assignable license to use the AIDS application software on the following computer:

Computer Manufacturer ______________________ Model __________

Serial Number ______________ O/S Release __________ 800 or 1600 BPI ________

This license does not include the right to reproduce, publish or license such program material to others for use on other computer systems. The Software Group, Inc. expressly reserves and the Licensee expressly consents that the entire right and title to the AIDS program materials shall remain with The Software Group, Inc. and that The Software Group, Inc. has the exclusive right to protect by copyright or otherwise, to reproduce, publish, sell and distribute such material to anyone. Licensee agrees to take all reasonable steps to ensure that the programs or any portion thereof, in any form, are not made available to any organization or individual not licensed by this agreement. The Software Group, Inc. warrants AIDS to be free from all software logic errors for a period of six (6) months. There are no other warranties, express or implied, arising out of or in connection with the use of or the performance of AIDS, including but not limited to, the implied warranties of merchantability and fitness for a particular purpose.

License Fee: $1195
Accepted by Licensee by ______________________

Title ______________________ Date ______________

## NO MORE WAITING FOR P. & L.'S

## IF YOU HAVE

**ADDS Mentor**

**DEC Ultimate**

**Honeywell Ultimate**

**IBM Series 1**

**Microdata**

**Prime Information**

**or any PICK operating system,**

this is the best* financial package you can buy. GAAP(TM) has a full warranty, complete documentation, password security control, and free upgrades for 1 year.

## GENERAL LEDGER

With GAAP(TM), you can have financial reports every day of the year. Year-to-date, period-to-date balance and budget analysis are a simple selection from the software menu. Capabilities include:

Chart of Accounts Maintenance • Journal Vouchers Entry • Recurring Journal Vouchers Entry • Journal Voucher Reports • Cash Receipts Entry • Trial Balance(s) • Statements (P & L, Bal/Sh., etc.) • Transaction Registers • Chart of Accounts Analysis (Audit Trails) • Chart of Accounts Listings • Statement Format Table Listings • General Ledger Summary Display • Suspended Batch Report • Summary Income/Expense Reports

## ACCOUNTS PAYABLE

Cash or accrual accounting, automatic check issuance, plus:

Voucher Entry and Inquiry • Voucher Registers • Cash Requirements Payment Schedules • Voucher Payment Selection Process • Issue Checks Process (Original issue) • Reissue Checks Process • Reset Check MICR Numbers • Manual Disbursements Posting • Void Checks Process • Cash Disbursements Journals • Check Registers • Check Reconciliation Process • Vendor Analysis (Audit Trail) • Vendor File Listings • Vendor File Maintenance • Bank Code File Maintenance • Terms File Maintenance

## ACCOUNTS RECEIVABLE

Produces ready-to-mail statements on a balance forward or open invoice basis. Functions include:

A/R Transaction Posting • A/R Payment-on-Account Application • Customer Open Invoices Listing • Customer Analysis • Aged Trial Balance • Finance Charge Calculation • Statement Printing • Delinquent Aged Trial Balance • Delinquent Statement Printing • Open Invoice Listing • A/R Transaction Registers • Zero Balance Invoice Removal • Customer File Maintenance

## OTHER OPTIONS

- Purchasing/Receiving Interface
- Inventory Control Interface
- Custom Software

## FREE TRIAL

We believe in our product so much that we want you to try it for 30 days. In fact, we guarantee a full refund, no questions asked, if within 90 days you determine GAAP(TM) does not meet your needs.

Turn your accounting system into a management tool, every day of the year! Call us today for more information.

**KDK** Enterprises, Inc.

Suite 302
1491 Chain Bridge Road
McLean, VA 22101
703/893-7883

*A Washington, D.C. auditing firm found that GAAP(TM), as the name implies, truly follows Generally Accepted Accounting Principles.

**GAAP** FINANCIAL SOFTWARE TM

```
SYSMAP.MENU
PQN
10 RO
T CLEAR,+
O 1 - Quit and logoff
O 2 - Exit to use the Terminal Control Language (TCL)
O 3 - Create or change file format data
O 4 - Create or change file cross reference data
O 5 - Create or change dictionary cross reference data
O 6 - Create or change program cross reference data
O 7 - Create or change proc cross reference data
O 8 - Create or change translate cross reference data
O 9 - Print file formats
O10 - Print file cross reference
O11 - Print dictionary cross reference
O12 - Print program cross reference
O13 - Print proc cross reference
O14 - Print translate cross reference
O15 - Print file descriptions
O16 - Print all cross reference reports
O17 - Print file format for a given file
O18 - Print file cross reference for a given file
O19 - Print dictionary cross reference for a given file
O20 - Print translate cross reference for a given file
O21 - Print proc cross reference for a given starting fragment
O22 - Print program cross reference for a range of names
O23 - Print proc cross reference for a range of names
OYour choice+
RI
IP?
T CLEAR
IF A = 1]2]3]4]5 GO 100]200]300]400]500
IF A = 6]7]8]9]10 GO 600]700]800]900]1000
IF A = 11]12]13]14]15 GO 1100]1200]1300]1400]1500
IF A = 16]17]18]19]20 GO 1600]1700]1800]1900]2000
IF A = 21]22]23 GO 2100]2200]2300
GO 10
100 HOFF
P
200 X
300 HRUN BP GET.FF
P
GO 10
400 HRUN BP GET.XF
P
GO 10
500 HRUN BP GET.XD
P
GO 10
600 HRUN BP GET.XB
P
GO 10
700 HRUN BP GET.XP
P
GO 10
800 HRUN BP GET.XT
P
GO 10
900 [SPL PRINT.FF]
GO 10
1000 [SPL PRINT.XF]
GO 10
1100 [SPL PRINT.XD]
GO 10
1200 [SPL PRINT.XB]
GO 10
1300 [SPL PRINT.XP]
GO 10
1400 [SPL PRINT.XT]
GO 10
1500 [SPL PRINT.FF.FILES]
GO 10
1600 [SPL PRINT.ALL.XREF]
GO 10
1700 [SPL PRINT.FILE.FF]
GO 10
1800 [SPL PRINT.FILE.XF]
GO 10
1900 [SPL PRINT.FILE.XD]
GO 10
2000 [SPL PRINT.FILE.XT]
GO 10
2100 [SPL PRINT.XP.FRAGMENT]
GO 10
2200 [SPL PRINT.XB.RANGE]
GO 10
2300 [SPL PRINT.XP.RANGE]
GO 10
```

```
Page 1 - Sysmap Report #14  - Translate Cross References - Printed 13:19:08  16 APR 1983

File...... Attribute...... AMC * Progs..... Procs................... Screens........ Files..... Words..............
                                 that       that                    that verify or  that       that
                                 translate  read/write/translate    translate       translate  translate

FF         AMC               1 A                                                    XF         AMC
                                                                                    XT         AMC
FF         AVS               2 A                                                    XT         AVS

***
```

```
Page 1 - Sysmap Report #18  - XF File Cross Reference - 13:16:59  16 APR 1983

File...... Attribute...... AMC Progs.............. Progs.............. Dictionary words.........
                               that                that                that
                               read (and use)      update              read

XF         FILE.ATR          0 GET.XF              GET.XF              AMC
                                                                       ATR
                                                                       FILE.NAME
XF         RPROG             1 GET.XF              GET.XF              RPROG
XF         UPROG             2 GET.XF              GET.XF              UPROG
XF         RPROC             3 GET.XF              GET.XF              WORDS
```

**SYSMAP Procs and Sample Reports**

## SYSMAP Procs Continued

```
   PRINT.ALL.XREF
01 PQN
02 [SPL PRINT.FF]
03 [SPL PRINT.XF]
04 [SPL PRINT.XD]
05 [SPL PRINT.XB]
06 [SPL PRINT.XP]
07 [SPL PRINT.XT]
08 RTN

   PRINT.FF
01 PQN
02 HSORT FF
03 H BY FILE.NAME
04 H BY AMC
05 H BREAK-ON FILE.NAME
06 H ATR
07 H AMC
08 H AVS
09 H DESC
10 H ID-SUPP
11 H HEADING "Page 'P' -
    Sysmap Report #9  -
    File Formats - Printe
   d 'TL'"
12 H LPTR
13 P
14 RTN

   PRINT.FF.FILES
01 PQN
02 HSORT FF
03 H WITH NO ATR
04 H FILE.NAME
05 H DESC
06 H ID-SUPP
07 H HEADING "Page 'P' -
    Sysmap Report #15  -
    File Descriptions -
   'TL'"
08 H LPTR
09 P
10 RTN

   PRINT.FILE.FF
01 PQN
02 100 HSORT FF
03 H WITH FILE.NAME
04 O
05 OFile name+
06 IP?
07 IF A = EX RTN
08 IF # A RTN
09 A"
10 H BY AMC
11 H BREAK-ON FILE.NAME
   "'BP'"
12 H ATR
13 H AMC
14 H AVS
15 H DESC
16 H ID-SUPP
17 H HEADING "Page 'P' -
    Sysmap Report #17  -
    'B' File Format - Pr
   inted 'TL'"
18 H LPTR
19 P
20 GO 100

   PRINT.FILE.XD
01 PQN
02 100 HSORT XD
03 H WITH FILE.NAME
04 O
05 OFile name+
06 IP?
07 IF A = EX RTN
08 IF # A RTN
09 A"
10 H BY WORD
11 H BREAK-ON FILE.NAME
   "'BP'"
12 H WORD
13 H PROCS
14 H WORDS
15 H ID-SUPP
16 H HEADING "Page 'P' -
    Sysmap Report #19  -
    'B' Dictionary Cross
    Reference - 'TL'"
17 H LPTR
18 P
19 GO 100

   PRINT.FILE.XF
01 PQN
02 100 HSORT XF
03 H WITH FILE.NAME
04 O
05 OFile name+
06 IP?
07 IF A = EX RTN
08 IF # A RTN
09 A"
10 H BY AMC
11 H BREAK-ON FILE.NAME
   "'BP'"
12 H ATR
13 H AMC
14 H RPROG
15 H UPROG
16 H WORDS
17 H ID-SUPP
18 H HEADING "Page 'P' -
    Sysmap Report #18  -
    'B' File Cross Refer
   ence - 'TL'"
19 H LPTR
20 P
21 GO 100

   PRINT.FILE.XT
01 PQN
02 100 HSORT XT
03 H WITH FILE.NAME
04 O
05 OFile name+
06 IP?
07 IF A = EX RTN
08 IF # A RTN
09 A"
10 H BY AMC
11 H BREAK-ON FILE.NAME
   "'BP'"
12 H ATR
13 H AMC
14 H AVS
15 H PROGS
16 H PROCS
17 H SCREENS
18 H FILES
19 H WORDS
20 H ID-SUPP
21 H HEADING "Page 'P' -
    Sysmap Report #20  -
    'B' Translate Cross
   Reference - 'TL'"
22 H LPTR
23 P
24 GO 100

   PRINT.XB
01 PQN
02 HSORT XB
03 H PROG
04 H PROCS
05 H ID-SUPP
06 H HEADING "Page 'P' -
    Sysmap Report #12  -
    Program Cross Refere
   nce - 'TL'"
07 H LPTR
08 P
09 RTN

   PRINT.XB.RANGE
01 PQN
02 100 RI
03 HSORT XB
04 H WITH PROG >=
05 O
06 OFrom name+
07 IP?
08 IF # A RTN
09 IF A = EX RTN
10 A"
11 O
12 OTo name+
13 IP?
14 IF A = EX RTN
15 IF # A RTN
16 H AND <=
17 A"
18 H PROG
19 H PROCS
20 H HEADING "Page 'P' -
    Sysmap Report #22  -
    Program Cross Refere
    nce From
21 B
22 A\
23 H to
24 A\
25 H - 'TL'"
26 H LPTR
27 H ID-SUPP
28 P
29 GO 100

   PRINT.XD
01 PQN
02 HSORT XD
03 H BY FILE.NAME
04 H BY WORD
05 H BREAK-ON FILE.NAME
06 H WORD
07 H PROCS
08 H WORDS
09 H ID-SUPP
10 H HEADING "Page 'P' -
    Sysmap Report #11  -
    Dictionary Cross Ref
   erences - 'TL'"
11 H LPTR
12 P
13 RTN

   PRINT.XF
01 PQN
02 HSORT XF
03 H BY FILE.NAME
04 H BY AMC
05 H BREAK-ON FILE.NAME
06 H ATR
07 H AMC
08 H RPROG
09 H UPROG
10 H WORDS
11 H ID-SUPP
12 H HEADING "Page 'P' -
    Sysmap Report #10  -
    File Cross Reference
   s - 'TL'"
13 H LPTR
14 P
15 RTN

   PRINT.XP
01 PQN
02 HSORT XP
03 H PROC
04 H PROCS
05 H ID-SUPP
06 H HEADING "Page 'P' -
    Sysmap Report #13  -
    Proc Cross Reference
    - Printed 'TL'"
07 H LPTR
08 P
09 RTN

   PRINT.XP.FRAGMENT
01 PQN
02 100 S1
03 O
04 OWhat is the starting
    fragment+
05 IP?
06 IF A = EX RTN
07 IF # A RTN
08 HSORT XP
09 H WITH PROC "
10 A\
11 H]"
12 H PROC
13 H PROCS
14 H ID-SUPP
15 H HEADING "Page 'P' -
    Sysmap Report #21  -
    Cross Reference for
   Procs Beginning with
16 B
17 A\
18 H - 'TL'"
19 H LPTR
20 P
21 GO 100

   PRINT.XP.RANGE
01 PQN
02 100 RI
03 HSORT XP
04 H WITH PROC >=
05 O
06 OFrom name+
07 IP?
08 IF # A RTN
09 IF A = EX RTN
10 A"
11 O
12 OTo name+
13 IP?
14 IF A = EX RTN
15 IF # A RTN
16 H AND <=
17 A"
18 H PROC
19 H PROCS
20 H HEADING "Page 'P' -
    Sysmap Report #23  -
    Proc Cross Reference
    From
21 B
22 A\
23 H to
24 A\
25 H - 'TL'"
26 H LPTR
27 H ID-SUPP
28 P
29 GO 100

   PRINT.XT
01 PQN
02 HSORT XT
03 H BY FILE.NAME
04 H BY AMC
05 H BREAK-ON FILE.NAME
06 H ATR
07 H AMC
08 H AVS
09 H PROGS
10 H PROCS
11 H SCREENS
12 H FILES
13 H WORDS
14 H ID-SUPP
15 H HEADING "Page 'P' -
    Sysmap Report #14  -
    Translate Cross Refe
   rences - Printed 'TL'
   "
16 H LPTR
17 P
18 RTN
```

Ⓟ

**Shebesta**

## SPECIALISTS—PICK SYSTEMS

**NEW • USED • BUY • SELL**

- 50 MB REFLEX DRIVES
- 10 MB DRIVES
- 16 K MEMORY
- 8 WAY BOARDS
- PRINTER CONTROLLERS
- PRINTERS
- PRISMS
- COMPLETE SYSTEMS
- ALTERNATIVES

**INTELLIGENT: • 8 WAYS • 16 WAYS • DISC CONTROLLERS**

**PRINTRONIX • 300 LPM $5950.00 & FREE VIEWPOINT • 600 LPM $7890.00 & FREE VIEWPOINT 60**

**USED 8000**
384 KB MOS
2-128 MB DISKS
24 PORTS
1600 BPI TAPE
600 LPM
PRINTRONIX
**$56,900.00**

**REFLEX I's USED**
IMMEDIATE DELIVERY

**REFLEX II's (NEW)**
MAY DELIVERY

**USED ULTIMATE "D"**
256 KB MOS
288 MB DISK
16 PORTS
800 BPI TAPE
PRNTR CONTR
**$60,975.00**

**USED REALITY 3.X**
64 K CORE
50 MB REFLEX
8 PORTS
800 BPI TAPE
PRNTR CONTR
**$13,900.00**

**USED SEQUEL!**
**1 MB MEMORY 24 PORTS, 1600 BPI**
**256 MB DISK, 4 CRTS, 300 LPM**
**SAVE $60,000!!!**

**WTB: ULTIMATE "C", DATA PRODUCTS, REFLEX DISK DRIVES**

**COMPUTERIZED CLASSIFIEDS—MIKEY DATA EXPRESS!**

**ON-LINE QUERY AND REGISTRATION OF WANTED TO BUYS AND WANTED TO SELLS**

| | | |
|---|---|---|
| LOGON | 513-232-5801 | 300 BAUD Q & R |
| MIKEY | 513-232-5802 | 1200 BAUD Q & R |
| | 513-232-5807 | recorder phone |

**513-232-5000**

## utilities

**Utility programs have saved many a programmer from the less appealing aspects of software development and maintenance. Reformatting code, stripping files clean of control characters, converting data from one format to another — all are examples of tasks best delegated to a program and not a programmer.**

**Good programmers will collect a "toolbox" of utility software to use from time to time. If you have a useful utility, send it in for publication. A regular feature in Pragma will be this Utilities Department, where good software tools will be spotlighted.**

# Uncompiling, Part 1

**The first in a series of articles devoted to uncompiling object code is presented. A program is provided to output object code as mnemonic opcodes for the hypothetical stack machine that the BASIC run-time system emulates.**

The DATA/BASIC™ compiler does not generate object code that can be directly executed by the hardware that the compiler is running on. Instead, the compiler generates object code for a hypothetical stack machine. When the object code is executed, an interpreter is invoked and executes instead. The interpreter emulates the stack machine by examining and interpreting the object code byte by byte, and performing the necessary operations encoded there. This approach to providing a high-level language for a computer can simplify implementation and allow the compiler to be relatively host-machine independent. The interpreted object code will typically execute much slower than if it could be directly executed by the host, but the stack machine technique has proved popular and is in use on many different brands of computer systems.

If the source code for a program is unavailable or has been lost or destroyed, the only way to reconstruct the program and guarantee it will still function exactly as before is to uncompile the object code. The program on the next page is the first step toward a complete uncompiler: it reads object code and reconstructs the equivalent stack machine operations in mnemonic form, similar to the output available when compiling with the A option on systems such as the ADDS Mentor. The program uses a file of opcode mnemonics shown below. Each item identifier is a two digit hex opcode, and attribute one is the mnemonic for each opcode. Only 144 of the 256 possible opcodes are shown, because both the opcode file and the uncompiler have been developed on a 3.2B Microdata™ using simple trial and error uncompilations of small programs for which the source code was known. Errors in the uncompiler (especially when handling addressing modes) and errors in the opcode file are undoubtedly present. Such errors can only be removed by knowing the full specifications for the generation of object code by the compiler, or through exhaustive decompilation tests with sample source programs. In the mean time, the present version of the uncompiler can successfully handle a surprisingly large percentage of existing object code.

Note that the uncompiler ignores four bytes before the precision specification in the object code. Those bytes have been found to contain a program's local and COMMON symbol table lengths. Also note that the PTR variable is the index to the next object code byte, BYTE is the same value but counting from one, and LOC is the final relative memory address location.

The next installment will address the problem of converting stack code back into BASIC statements. P

```
00 NULL           01 END OF LINE     03 LOAD ADDR       05 PUSH            06 GO TO           10 GO IF ZERO      11 GO IF NONZERO
12 FOR TEST       13 GOSUB           14 RETURN TO       20 PUSH            22 PUSH ADDR       24 POP             30 PUSH MAT ADDR
32 PUSH MAT ADDR  34 POP MAT         40 PUSH            62 LOCK            63 UNLOCK          66 MAT INDEX       67 MAT INDEX
68 MAT ASSIGN     69 MAT COPY        6E INPUT USING     70 INPUT           71 INPUT,_         72 INPUT,          73 INPUT:
74 INPUT,:        75 INPUT,:_        76 PRINT:          77 PRINT           78 PRINT NOTHING   79 PRINTER CNTRL   7A PRINT ON
7B PAGE           7C HEADING         7D FOOTING         7E TAB             7F PROMPT          80 READC           82 READ
83 READU          85 READV           86 READVU          88 WRITE           89 WRITEV          8A MATREAD         8B MATREADU
8C MATWRITE       8D SELECT          8E READNEXT        90 OPEN            91 DELETE          92 CLEARFILE       94 RELEASE
95 READT          96 WRITET          97 WEOF            98 REWIND          9C PROCREAD        9D PROCWRITE       9E SHARE
A0 AND            A1 NOT             A2 OR              A3 <               A4 <=              A5 =               A6 MATCH
A8 PUSH 1         A9 +               AA -               AB *               AC /               AD 0-              AE :
AF PUSH 0?        B0 STORE INDIRECT  B1 ON POP GO TO    B2 ON POP GOSUB    B3 SUBROUTINE      B4 CHAIN           B5 ENTER
B7 CALL           B8 CALL INDIRECT   B9 BREAK KEY ON    BA BREAK KEY OFF   BB DATA            BC RQM             BD CLEAR
BF RETURN         C0 ERRMSG ID       C1 STOP            C2 MARK ARGS       C3 DEBUG           C4 DELETE2         C5 EXTRACT2
C6 DELETE1        C7 EXTRACT1        C8 INSERT2         C9 REPLACE2        CA INSERT1         CB REPLACE1        D0 DATE
D1 TIME           D2 TIMEDATE        D3 ICONV           D4 OCONV           D5 NUM             D6 DELETE3         D7 EXTRACT3
D8 INSERT3        D9 LOCATE ATR      DA REPLACE3        DB ABS             DC INT             DE MOD             DF PWR
E0 RND            E1 SQRT            E2 LN              E3 EXP             E4 COS             E5 SIN             E6 TAN
E7 @              E8 ASCII           E9 CHAR            EA COL1            EB COL2            EC COUNT           ED EBCDIC
EE FIELD          EF FORMAT          F0 INDEX           F1 LEN             F2 SEQ             F3 SPACE           F4 STR
F5 []             F6 TRIM            F8 LOCATE SUBVAL   F9 LOCATE VALUE
```

```
EQU PWR3 TO 4096, PWR4 TO 65536, EOL TO CHAR(1)
PRINT "FILE": ; INPUT FILE
OPEN FILE ELSE STOP "NO SUCH FILE!"
PRINT "ITEM": ; INPUT ITEM.ID
READ ITEM FROM ITEM.ID ELSE STOP "NO SUCH ITEM!"
OPEN "OPCODES" ELSE STOP "NO OPCODES!"
DEL ITEM<1> ; FIRST.LINE = ITEM<1>
PRECISE = CHAR(SEQ(FIRST.LINE[5,1])+48)
IF PRECISE # 4 THEN PRINT "PRECISION ":PRECISE
ITEM<1> = FIRST.LINE[6,LEN(FIRST.LINE)-5]
TOTAL.LINES = COUNT(ITEM,EOL)
IF TOTAL.LINES = 0 THEN TOTAL.LINES = 1
LOC = 46 ; BYTE = 1
HEADING "PAGE 'P' - UNCOMPILATION OF ":FILE:" ":ITEM.ID:" - 'TL'"
FOR I = 1 TO TOTAL.LINES
  INSTRUCTS=FIELD(ITEM,EOL,I):EOL
  PRINT
  PRINT " BYTE   LOC   HX   LINE ":I:" ":STR("-",10)
  PRINT
  PTR = 1
  LOOP
    OPCODE=INSTRUCTS[PTR,1]
  UNTIL OPCODE="" DO
    HEXCODE=OCONV(OPCODE,"MX")
    READ SOURCE FROM HEXCODE ELSE SOURCE = "OPCODE ":HEXCODE
    PRINT BYTE "R#5" : LOC "R#6" : HEXCODE "R#5" :
    PRINT "   " : SOURCE : " " :
    PTR = PTR+1 ; LOC=LOC+1 ; BYTE = BYTE+1
    GOSUB 100 ;* DECODE
    PRINT
  REPEAT
  PRINT
NEXT I
STOP
*
100 * DECODE
BEGIN CASE
CASE (HEXCODE="03") OR (("20"<=HEXCODE) AND (HEXCODE<="24"))
  GOSUB 500 ;* PRINT ADDR
CASE (HEXCODE="30") OR (HEXCODE="32") OR (HEXCODE="34")
  GOSUB 500 ;* PRINT ADDR
  HEX.STRING = INSTRUCTS[PTR,4]
  PTR = PTR+4 ; LOC=LOC+2 ; BYTE = BYTE+4
  GOSUB 400 ;* HEX TO DEC
  PRINT "(":DEC.VALUE:",":
  HEX.STRING = INSTRUCTS[PTR,4]
  PTR = PTR+4 ; LOC=LOC+2 ; BYTE = BYTE+4
  GOSUB 400 ;* HEX TO DEC
  PRINT DEC.VALUE:")":
CASE (HEXCODE="05") OR (HEXCODE="55")
  LOOP
    NEXT.BYTE = INSTRUCTS[PTR,1]
    PTR = PTR+1 ; BYTE = BYTE+1
  UNTIL NEXT.BYTE = CHAR(254) DO
    PRINT NEXT.BYTE:
  REPEAT
  LOC = LOC+6
CASE (HEXCODE="06") OR (("10"<=HEXCODE) AND (HEXCODE<="14"))
  GOSUB 200 ;* PRINT OFFSET
CASE HEXCODE = "40"
  PRINT "'":
  LOOP
    NEXT.BYTE = INSTRUCTS[PTR,1]
    PTR = PTR+1 ; LOC=LOC+1 ; BYTE = BYTE+1
  UNTIL NEXT.BYTE = CHAR(254) DO
    PRINT NEXT.BYTE:
  REPEAT
  PRINT "'":
CASE (HEXCODE="B1") OR (HEXCODE="B2")
  LOOP WHILE OCONV(INSTRUCTS[PTR,1],"MX") # "C1" DO
    PTR = PTR+1 ; LOC=LOC+1 ; BYTE = BYTE+1
    GOSUB 200 ;* PRINT OFFSET
    PRINT " ":
  REPEAT
END CASE
RETURN
*
200 * OFFSET
HEX.STRING = INSTRUCTS[PTR,4]
PTR = PTR+4 ; LOC=LOC+2 ; BYTE = BYTE+4
GOSUB 400 ;* HEX TO DEC
IF DEC.VALUE > PWR3 THEN DEC.VALUE = DEC.VALUE-PWR4
PRINT LOC+DEC.VALUE-1:
RETURN
*
400 * HEX TO DEC
DEC.VALUE = 0
FOR H = 1 TO 4
  DIGIT = HEX.STRING[H,1]
  IF DIGIT >= "A" THEN DIGIT = SEQ(DIGIT)-55
  DEC.VALUE = DEC.VALUE + (DIGIT*PWR(16,4-H))
NEXT H
RETURN
*
500 * PRINT ADDR
ADDR = INSTRUCTS[PTR,4]
PTR = PTR+4 ; LOC=LOC+2 ; BYTE = BYTE+4
PRINT "V":ICONV(ADDR,"MD0"):
RETURN
*
END
```

## UNCOMPILER
## LISTING

Ⓟ

## user profile

**Are all data processing installations the same? How do managers actually manage, how do programmers program, how do operators operate, how do users use their data processing systems? What defines the leading edge of current, modern information processing?**

**In this regular department, Pragma will be interviewing personnel at a variety of installations, to reveal the who, what, when, where, why and how of actual data processing organizations.**

**The software package called Information provides Prime computer users with capabilities that are more or less compatible with other Pick-style systems. Pacific Valley Bank, a seven year old San Jose based firm of over 400 employees, is a user of Prime hardware and Information software. One department of the bank is the Information Systems Group, which provides timesharing for its clients, much like other data processing service bureaus. For this issue's user profile, Pragma interviewed Alexander Lange, System Administrator for the Pacific Valley Bank Information Systems Group.**

**Pragma:** Why did Pacific Valley Bank discontinue using Microdata™ hardware and convert to Prime?

**Lange:** Our customer base had grown very quickly and was on the verge of growing again very quickly, nearly half again as many users as we had on the Microdata. On the Microdata we had thirty active ports including the spooler, and we were just at the limit on that machine. Also, we had intended to branch out into other kinds of software products, to provide to our users not just products running under BASIC or under the Pick operating system, but also FORTRAN packages and things of that nature.

**Pragma:** In your opinion, what are some of the major advantages of Prime over Microdata?

**Lange:** The Information version of the Pick operating system is much more user friendly than the Reality operating system. It will tolerate many more sloppy errors on the part of someone typing in commands. Say, if you happen to call up its editor with an erroneous file name, it will tell you that and ask what is the proper file name. Which is a very useful thing, if you think about it. Say you have just gone through some large SELECT process and have an active SELECT list of many items that took you quite a while to pull out of a file. If then you happen to type in a wrong file name, you have a tendency to pull your hair out. In Prime Information, you also have not only one SELECT list active at any one time, but up to ten active lists that you can manipulate in different ways. Also, all of the commands entered are stored in a stack, which you can then edit and change and re-execute in different ways. Very repetitive commands that you may have to keep typing in can be recalled and executed with two keystrokes. There's a processor that allows you to prestore very sophisticated editor commands — much more sophisticated than the Microdata editor. With certain commands that weren't provided in DATA/BASIC™, you can do some very fancy work with Prime's INFO/BASIC. Prime also supports a wide number of communication protocols. It can be networked over a long distance to other Prime machines.

**Pragma:** How about disadvantages?

**Lange:** The quality of the hardware that we had at the time of the Microdata, say the tape drive, left a lot to be desired. The Prime is a much newer machine than we had at that time, so it's a little bit hard for me to come down with disadvantages about the Prime. I'd say the disk I/O on the Prime has always been a bottleneck. We have a response time problem that is considerably more serious than it was on the Microdata. The Prime Information system uses fourteen different hashing algorithms for allocating disk space for file records. The size and number of different types of files on the Prime is considerably greater, which is an advantage in some ways, but then it makes the file maintenance tasks much more difficult, in so far as the number of different things you have to check in order to select the proper modulos. So disk segmentation is a bad problem on a Prime, unless one has the opportunity in the normal course of work to stay on top of resizing jobs. It's quite easy to let your files get out of hand and cause response time problems.

**Pragma:** Tell us more about your disk I/O bottleneck.

**Lange:** Right now we are running two 300 megabyte storage modules, with only one disk control unit. As a result, the CPU is running along very nicely, but it is continually waiting for data off the disk. All of our applications rely very heavily on SELECTs and large groups of data out of different files that all have to be called up off of the disk. With only the one control unit, we're constantly waiting on all of our jobs. That will be rectified by the purchase of another control unit, but still I'm told that you reach an upper limit of about 45% in disk I/O as the maximum that you can achieve as far as performance, in getting the data off the disk and through the programs and back out to disk. This is what I've learned in my experience over about the past ten months, and in talking with other people: there has always been a disk I/O bottleneck on Prime.

**Pragma:** What do you miss about the Microdata?

**Lange:** Very little, really. The Microdata was a good machine, but the Prime just really has it all over the Microdata. When I think back to the Microdata, all I get are pictures of all night sessions with the machine, trying to boot it from tape and not having a good tape, the tape drive not working properly, and me being stuck with just a chunk of metal that won't work because I can't get the operating system onto it from the tape. On the Prime, you boot from disk with three simple commands and then you have your full system ready to go. With the Microdata, we continually had problems with it. I guess it maybe was improperly maintained. At that time, we really had no one who was responsible for the device.

**Pragma:** Do you think you happened to have a lemon?
**Lange:** No, I don't think so. I think it was really that the machine was pushed very hard, and there was no one who was administering it properly as far as, say, the files and the file hashing, so that the files on the Microdata were very poorly allocated. The machine was really taxed to its limits. I would like to think I didn't have a lemon.

**Pragma:** Describe your Prime configuration.
**Lange:** Our initial Prime configuration was the Prime 750 CPU itself, with two megabytes of main memory, and we have a 75 inch per second dual density Kennedy tape drive on that. One 300 megabyte disk storage module, and that was back about ten months ago. Since that time, we've added another 300 megabyte storage module, and two more megabytes of main memory, so we now have four megabytes of memory. We have twelve multiplexors as the core to our data communications network, one system line printer, a 600 line per minute Printronix that we got through Microdata, but since that time the Microdata nameplate has fallen off. We have, out in the field, approximately 65 CRTs and half that number of slave printers running off of those terminals. We run all of our terminals at 1200 baud. Right now the current limit on most of our multiplexors is 1200 baud per port, so that's how we have all of our CPU lines configured. Our users are all hooked into that machine through our data communications network. Our customer base stretches all over the state of California and we've just gone interstate about two months ago. Our customer base is going to continue to grow, and I know that we will be expanding either this machine or getting another machine to go along with this one, but how soon that will be I don't really know.

**Pragma:** How many more terminals do you think you could put on without changing anything else?
**Lange:** We could put on about sixteen more terminals. We wouldn't want to get much above supporting sixteen more. You reach an upper limit of 96 users, above which your performance problems are too great. Prime says the machine is the 128 user version, but it's kind of known that you really shouldn't get above 96 users.

**Pragma:** Prime seems to heavily promote their networking capabilities. How is having two Prime Information systems sharing the same database better than simply having one bigger more powerful system with more terminals?
**Lange:** Beyond the Prime 750, the next step up is the Prime 850, which is actually two Prime 750 machines that have been yoked together to work in conjunction with one another. If we were to stick with Prime, we could not expand except one step up beyond this. So the concern is not so much bigger devices. The concern is about more of the same type of device to handle our increasing customer base.

**Pragma:** Are you getting the impression that Prime is giving the Information system a lot more attention lately?
**Lange:** Yes, considerably more. Of course, my experience doesn't extend back that far with Prime Information. But we have recently gone through an upgrade in the Information system. We're running Information release 5.2. We were on 5.1. At 5.2, more than one active SELECT list came about. They tightened up Information's interface with the Primos operating system. In file opens, file lookups, it's a little bit faster now. So it does seem, yes, that the Information group of Prime is maybe being taken a little bit more seriously now.

**Pragma:** Describe the conversion that took you from Microdata to Prime.
**Lange:** We had quite a number of applications that we had to bring over from the Microdata, and they all came over with relative ease, depending on the complexity of the application and its value to management at that time. Our most successful application was brought over to the Prime in the space of about a week and a half. We worked with a consulting company. They had been doing conversions from Microdata and a number of machines to Prime, and they were very expert at it. They already had their conversion programs written from previous conversions, and they handled the whole conversion. It was very smooth. One of our applications did have some difficulty because we had design problems with it. It was a less better defined application than the earlier ones, our more successful products, so the conversion there was bumpy. But all in all it was very smooth.

**Pragma:** What was the total length of time from the day you were only on the Microdata to the day you were only on the Prime?
**Lange:** The total length of time was about six months. When I came on board in May of '81, they had just recently got the Microdata Reality® 8000. Before that they had a Reality 6000. We kept the Microdata in production until December of '82, but the Prime arrived in May of '82, so that from May to December '82 we had the two machines running in parallel. We had our user base split off on both machines for about four months of that time. The reasons for that were the delays that we faced in the conversion of the last applications. At that time we didn't have sufficient storage capabilities on the Prime, so before we could finally get our last users off the Microdata, we had to very quickly purchase another 300 megabyte storage module in order to accommodate them.

**Pragma:** Before you began, how long was the conversion expected to take?
**Lange:** I would say it was intended to take approximately three months. One problem or another kept delaying us until finally it just became absolutely critical, and we made a big push and got off the Microdata. So I'd say it was about twice as long as we expected.

**Pragma:** The Prime allows the user to type ahead in answer to a future program prompt that has not yet actually occurred. Describe the difference in the operator interface now that type ahead is available.
**Lange:** I'd say it's more ergonomic. It's more pleasing to the operator to have that ability. Of course, there's an initial period that a person goes through in getting used to type ahead. It can be frustrating to type ahead of an error, but there are safeguards in the operating system to reprompt you for file names and things like that. It does take a little getting used to. It's actually quite nice. You may have a number of jobs that you would like to execute through the night, without you being there in attendance to put in the next command. You can type ahead all those commands and go home and sleep, then come back in the morning and you'll have your report there. Say you had many different jobs, and you didn't have a proc or programs built so you could just invoke the one master program and let it all run. If you just know all the sequences, you can type them all ahead.

**Pragma:** The "real" operating system for the Prime machine is Primos, and Information is essentially running as a giant application program as far as Primos is concerned...
**Lange:** Yes, as FORTRAN programs running under the Primos operating system.

**Pragma:** What kind of good and bad things does that mean for the user?

**Lange:** I guess you're talking about the idea that there's always an extra level. It's less of a direct relationship with the host operating system. I don't know if there's a measurable delay or increase in response time because of that additional step in the interface to the operating system, but Information is a very fast set of programs. From the operator point of view, it's no slower than just using the operating system "directly", such as when using a Microdata machine.

**Pragma:** Do you find yourself using more and more of the Primos facilities, or are you restricting yourself to just the Information capabilities?

**Lange:** Well, I rely every day on Primos operating system utilities, such as when we back our system up every night. We also use the file utility manager, which is a very powerful tool for moving data around on the disk. There are certain guidelines however, for when you are going to be handling or manipulating Information level files. Information supplies you with all the tools necessary so that the files are handled properly. Sometimes, on earlier releases, Primos didn't take into consideration the special features of the Information data files, and often it could damage them. Or, through the misuse of Primos level commands, you could create some problems with your database, but that's really through inexperience. What I'm trying to say is that, except for system backup, Information supplies you with all that you need to handle Information files.

**Pragma:** Do you think that, because of the difference in Prime Information compared to other Pick-like operating systems, you're committing yourself to staying with Prime forever? Or do you think you'll always have the choice to convert to some other system?

**Lange:** I guess that would depend on how much our entire application product line had a tendency to become more tightly interfaced. If we were to try to be more integrated under Primos, then I guess we would have to rely heavily on Prime and never think of leaving. Many shops develop their applications to be carried across from machine to machine, and that has its own special disadvantages as well. You are never fully utilizing the capabilities of any one single machine. Certain different machines handle different types of processing in some very good ways that really speed up processing time. You can never take advantage of that if you try to make your products transportable across devices. So that's a really tough decision. I think with Prime Information, we'd always have the choice of being able to transport it to another machine, because it is Pick based, but it would require some tailoring unless we designed the products with that in mind.

**Pragma:** Describe all the applications you have running.

**Lange:** The majority of them are accounting applications. We run a trust accounting system for agencies at the county level around the state. We also run applications for different departments of the bank itself: applications to track pooled certificates of deposit and other investment money for the bank's customers. We run bookstore inventory control software and accounting software. We run a word processing application. We also have a financial modeling system. It does not run under Information, but nevertheless it runs on our Prime machine.

**Pragma:** How is security handled on your system?

**Lange:** All of our remote peripheral devices are kept at our user sites. The majority of them are in security guarded areas at the county level and on university campuses, so we really rely on our users' own security measures to protect access to the peripherals themselves. Entry into the system is guarded at the Primos level with a password, and at the Information level with a password, and at the applications level with many internal passwords, so that we have three levels of password security for entry into the Prime system itself. We have restricted dialup lines, and very few people know about those numbers, and then they would still have to face the Prime password security. So we don't really have great concern about theft or misuse of our system. We have a guarded computer facility, with a guard on duty 24 hours a day. He will allow only certain individuals who have been identified to him into the security guarded area. The actual system hardware is right next to the guard station, and any interlopers would be easily apprehended.

**Pragma:** Do you think that if someone logged on as a customer, but they knew a little about Prime Information and Primos, that they could access parts of the system you wouldn't want them to?

**Lange:** Yes, I do, but I think that's probably true with any system you would care to name. I wouldn't know the particular security problems on particular machines, but they all have one drawback. That is if somebody smart enough wants to do some damage or wreak some havoc, they can do it. Now at our user sites, our customers are not computer intelligent, nor do they allow the possibility of computer intelligent people getting near them. Our devices are running in county offices, and counties typically can't pay the kind of money for the sophisticated kind of data processing staff that would cause a security problem for us. But certainly anybody intelligent enough or with enough experience could get into the system and do things that we wouldn't want them to do.

**Pragma:** Do you think it would be particularly difficult for someone to access your system and then clean any evidence showing that they were there, so that they could essentially get free computer time and make it difficult for you to detect that they are using the system?

**Lange:** The system records all logins and all logouts, and produces a hardcopy automatically at login, not like the accounting that's provided on the Microdata Reality, so we always have a record of all logins.

**Pragma:** Is there anything on your personal wish list that you'd like to see improved in Prime Information?

**Lange:** There are certain limitations, such as having to rely on the SELECT processor quite a bit, but that can be gotten around with application design changes to rely on stored lists instead of constantly having to retrieve the same records off the disk. In Primos, the magnetic tape backup utility has a limitation on blocking. For some reason, you can only block a tape up to 2048K. I don't know why that limitation is there. But beyond those picayune statements, there's not much I would like to change right now. We have just upgraded to 5.2 Information, so we're still really getting familiar with it.

**Pragma:** Did you have any special expectations prior to acquiring the Prime, and how have your expectations been met?

**Lange:** The Microdata Reality system was the first data base management system I had ever worked on. Over the period of time that I worked on it, I grew quite fond of it, and was not looking forward to moving away from it. But then, I looked upon it as a challenge. When I started getting a feel for the Information system, I found myself no longer dreaming back to the days when I worked on the Microdata. I've been enthralled by Prime Information ever since. P

## benchmarks

**Are you thinking of doing an upgrade to your hardware or software? Are you comparing throughput and performance while shopping around for a system? Have you converted from one machine to another? Be sure to send in your benchmark statistics to Pragma, so the results can be featured in this department and shared with other installations.**

# Left vs. Right Justification

**Sorting with right justification can take 50% more time than sorting with left justification.**

If a file of five-digit ZIP codes is sorted using left justification, is there any difference in speed compared to sorting the same file with right justification? To find out, this seven line proc

```
    BENCH
001 PQN
002 HTIME
003 P
004 HSSELECT ZIP
005 STON
006 HTIME
007 P
```

was tested with a Reality® on a file of items named ZIP. Each item in the ZIP file consisted of a city and state name, with every item having a five-digit ZIP code item identifier. The sort of the file was timed twice, once by running the proc with the justification code in the file's DL/ID item set to "L", and again (after being careful to flush all memory buffers of residual data) with the DL/ID justification code set to "R". The results are shown in the first row of the table below.

After the first two runs, the file was then enlarged by recreating the file with a bigger modulo, and concatenating each item identifier with itself (doubling its length three times) until each identifier was 40 digits long, even though other attributes in each item remained the same. The file was again sorted two more times, once with left justification and once with right justification. The results form the second row in the table below.

The timings obtained are very interesting. Since every item identifier in the file has the same number of digits, sorting each identifier from left to right or from right to left produces the same ordering of items, but sorting with right justification appears to require 50% more time! Also, longer identifiers do not increase the difference in sort time (in fact, it was reduced in the tests shown here), implying that there is a fixed penalty for right justification independent of the data being sorted.

The lesson here is obvious: if every item in a file has the same number of identifier characters, avoid sorting with right justification. If the file has numeric identifiers with varying lengths that require right justification for sorting, consider padding all of the identifiers (say, with zeros) to one standard maximum length so that left justified sorting can be used. However, beware that trying to predict what the maximum number of digits for a data item should be can have serious consequences, such as causing increased use of storage for pad characters, and forcing a growth limit on the file. (If a file of check numbers is padded to only five digits, what happens when check #100,000 needs to be input?)

The results presented here are convincing enough to show that right justification imposes a major execution time penalty, but the tests also suggest a number of new questions. Do similar results occur on machines other than Reality? Does the penalty for right justification always decrease as identifier length increases as shown here, or is that difference not really significant? Are results the same when explicitly sorting by a dictionary word, instead of implicitly sorting by the DL/ID item? Are results the same when sorting by some arbitrary attribute instead of by the item identifier? All of these questions could use further investigation. P

| | Modulo | Items | Bytes | DL/ID = L | DL/ID = R | Increase |
|---|---|---|---|---|---|---|
| Test 1 and 2: | 1,109 | 24,687 | 554,089 | 660 sec | 1,047 sec | 59% |
| Test 3 and 4: | 2,837 | 24,687 | 1,418,134 | 2,169 sec | 3,294 sec | 52% |

## wish list

**Users of computer systems should always remember that the nice thing about software (besides the fact that it doesn't break when you drop it) is that programs are relatively easy to change — no soldering gun is necessary. So just because the operating system or the compiler or a system utility happens to work (or fail to work) in some particular fashion now, doesn't mean it has to always be that way. The next time an inspired idea arrives for improving your system, write it down and send it to Pragma for publication in the Wish List. Naturally, all such submittals are eligible under Pragma's author payment program.**

*The previous 61 Wish List items have been featured in issues #1 through #3 of Pragma.*

**62. COPY Option to Suppress Null Attributes.** Often a data item will have a number of attributes that are null and that therefore provide no additional information when included in output generated with the COPY verb and the (P) or (T) options. Provide another option for COPY so that the attribute number and blank line for a null attribute is suppressed on such output, thereby condensing items to only show non-null data.

**63. Default Attribute Names.** [Mark A. Johnson, C.F. Guyon Inc., Harrison NJ] Allow ENGLISH® to automatically recognize command words of the pattern "*A'0N"' and to then include the corresponding attribute data in the usual left justified 10 character column. That way any attribute number can quickly and easily be referenced and listed, while dictionaries and LISTDICTS output do not have to be cluttered with a lot of standard definitions.

**64. Aborting Editor Commands.** Often a user needs to abort a currently executing editor command (such as a long list or multiple replace command) without exiting the editor and losing all edits already made to the item. Provide a mechanism to allow an editor command to be aborted, with normal editor input then resuming. (This wish has been recently re-implemented on Prime Information.)

**65. Correcting Typed Commands.** Operators frequently type long commands only to discover a typo far from the current cursor position. To fix the typo requires retyping the command or using the backspace key to move the cursor back while destroying much of what was typed. Provide non-destructive cursor movement keys to move the cursor left or right within the command being typed without erasing any of the command. This allows the cursor to be repositioned over typos which can then be corrected by overstriking. Note that it should be allowable to then hit the RETURN key while the cursor is imbedded in the middle of a command. P

## A Comparison of BASIC Implementations

**The results of a survey comparing the BASIC compiler implementations of four major Pick-style operating systems are shown in a table designed to highlight differences and incompatibilities among the four products.**

Early this year, a preliminary edition of the Cosmos R/BASIC reference manual became available. Inspection of the manual revealed that R/BASIC is in many ways similar to other Pick-style BASICs, but it is also different in many ways. Just how similar to one another are these BASICs? A survey was begun, and the results are presented in the table beginning on the opposite page. Of the many Pick-oriented BASICs on the market, four were chosen as the basis for the survey. Besides inspiring the survey, the Cosmos implementation was chosen for comparison because it is a single-user, inexpensive, unbundled version available for a microcomputer that has already been shipped in huge quantities. The Prime Information version of BASIC was chosen because it was the first "reverse-engineered" implementation and is generally recognized as offering close to a superset of the features of competing systems. The Microdata™ implementation was included since it is the most widely installed and was the first of all versions. And last but not least, the ADDS implementation was chosen because it well represents the version more or less currently being licensed by Pick to a number of vendors.

In order to create the table presented here, documentation for each of the four systems was obtained, and descriptions of every BASIC statement and major language feature in the documentation were compared. Actual computers were not used to test reported differences or similarities. Also, only features of the BASIC language itself were compared. Compile-time options, debuggers, commands for executing object code, error messages and other such environmental aspects often covered in the reference manuals were not included in the survey. If a given statement or feature (for example, the ABS function) was found to be identical in all four implementations, it was excluded from the table. If a feature was unique to one implementation, or missing entirely while found to be identical in the remaining *three* implementations, then the item is described under Unique Features at the end of the table. Otherwise, the item of interest is supported in a variety of ways and so is described in a row in the table, which is ordered more or less alphabetically. Whenever possible, the "greatest common factor" of a language feature (often simply described in the table as "available") was chosen as a standard for each table row, while a more descriptive explanation was included if any implementation differed in some way. For example, three of the implementations offer an ABORT statement which is supported in the simplest form by Cosmos and Prime, but which is available with an added feature under ADDS. Also note that because of the number and complexity of the conversions supported by ICONV and OCONV, they are not compared here in detail.

Thanks go to Roger Harpel and Dave Ostby of Cosmos for supplying manuals and last minute updates, to Allison Hartmann of IDS for sending the Prime manual, to Dave Harrar of Computyme for providing the updates describing changes in

Prime's latest releases, and to Carol Tomlinson and Dave Yulke of ADDS for supplying a manual and updates. Prime easily wins first place honors for best documentation. Their reference manual is well organized (mostly because of keeping things alphabetical, although statements and functions are segregated), and features are described in detail. Cosmos keeps the same good organization, making the absence of an index no big problem, although the explanations are not as thorough as Prime's. The new $30 Cosmos manual is a bit of an improvement over the old $20 preliminary edition, being more complete and fixing some amazing English, such as "disgression" for "discretion", "strickly" for "strictly", and "syntaxably" for "syntactically"! The ADDS documentation trails far back in last place, being poorly organized (apparently as a result of attempts to cut and paste the initial Microdata-style reference manual into a kind of tutorial) and often lacking in detail (see this article's comparison of the COUNT function, for example). P

## A Comparison of BASIC Implementations

| | Cosmos | Prime | Microdata | ADDS |
|---|---|---|---|---|
| **ABORT statement.** | Available. | Available. | Not available. | Optional expression list is used to output ERRMSG items. |
| **BREAK statement.** | KEY clause is optional. Expression is allowed instead of ON/OFF. ON or OFF either sets or clears a break flag. Flag is ON at start of program execution. | KEY clause is optional. Expression is allowed instead of ON/OFF. ON and OFF decrement and increment a break inhibit counter. Counter is left unchanged by program initialization or termination. | KEY clause is required. ON or OFF either sets or clears a break flag. Flag is cleared at program termination. | No KEY clause. ON and OFF decrement and increment a break inhibit counter. Program termination state is not specified by documentation. |
| **ATAN function.** | Available. | Available. | Not available. | Not available. |
| **CLEAR statement.** | Initializes all non-COMMON variables (or all variables if COMMON clause is included) to null. | Initializes all non-COMMON variables (or all variables if COMMON clause is included) to zero. | Initializes all variables to zero. | Initializes all variables to zero. |
| **CLEARDATA, CLEARSELECT statements** to clear DATA and SELECT lists. | Available. | Available. | Not available. | Not available. |
| **CLEARFILE statement.** | Not available. | Available. | Available. | Can't clear dictionaries. |
| **COL1, COL2 functions.** | Can return substring delimiter positions found when using search feature in [ ] substring extractions. | Values returned are local to program or subroutine being executed. | Error if no previous FIELD executed. COL2 returns string length plus one if delimiter not found. | Error if no previous FIELD executed. COL2 returns string length plus one if delimiter not found. COL2 equals COL1 plus 1 if delimiter is null. |
| **CONVERT statement** to change one set of characters in a string to another set. | Available. | Strings cannot be longer than 128 characters. | Not available. | Not available. |
| **COUNT function.** | COUNT("AAAA","AA") equals 2. | COUNT("AAAA","AA") equals 2. | COUNT("AAAA","AA") equals 3. | Overlap rule not specified. |
| **DEBUG statement.** | Not available. | Not available. | Available. | Available. |
| **DELETE, EXTRACT, INSERT, REPLACE functions and statements.** | Arguments are optional when used in angle bracket form (EXTRACT and REPLACE functions only). | Arguments are optional when used in angle bracket form (EXTRACT function only). | Arguments are optional when used in angle bracket form (all four functions). DEL and INS statements allow deletions and insertions without assignment statements. | Arguments are optional in parenthetic function or angle bracket form (EXTRACT and REPLACE functions only). |
| **DIMENSION statement.** | Dimensions may be defined by an expression. An additional "zero element" is allocated for each matrix. | Dimensions may be defined by an expression. An additional "zero element" is allocated for each matrix. Dimensions may be changed during program execution. | Available. | Available. |

| | Cosmos | Prime | Microdata | ADDS |
|---|---|---|---|---|
| **ECHO statement** to control display of input. | Available. | Not available. | Not available. | Available. |
| **EQUATE statement.** | Only one symbol may be equated per statement. The equated value may be a function. Two dynamic array delimiters are predefined equate symbols prefixed by@. | LITERALLY or LIT may be used for TO, in which case the literal must be a string and surrounding quotes are significant. The equated value may be a function. One EQUATE statement may be broken into multiple lines. Four dynamic array delimiters are predefined equate symbols prefixed by @. | Available. | Available. |
| **EXECUTE, PERFORM statements** to invoke any system command and then resume program execution. | SELECT lists are stacked and reinitialized when EXECUTE version is used. | Print files, SELECT lists, execution and record locks and the BREAK key are not reinitialized. | Not available. | Not available. |
| **FIELD function.** | Optional fourth argument indicates number of fields to return. | Optional fourth argument indicates number of fields to return. | Available. | Available. |
| **FMT function.** | Format string is: justification (C,L,R or T), mask (any characters with # for digits), field size (optional integer). | Format string is: field size (optional integer), background (optional pad characters), justification (L,R or T), conversion (optional number of fractional digits, $, comma, zero suppress), mask (optional non-numerics with # for digits). STATUS function returns conversion status. Maximum length of result is 188 characters. | Not available. See format specifications. | Not available. See format specifications. |
| **GARBAGECOLLECT statement.** | Not available. | Available. | Ignored. | Not available. |
| **GOSUB, GOTO, RETURN TO statements.** | Trailing colon is optional. | Trailing colon is optional. | Available. | Available. |
| **HEADING, FOOTING statements.** | Not available. | ON clause allowed. Page number can't appear in both HEADING and FOOTING. | PP control allowed. | C control allowed. |
| **ICONV, OCONV functions.** | STATUS function returns conversion status. D, HEX,MD,MT,MX conversions specified. | STATUS function returns conversion status. Input cannot exeed 188 characters. D,MB,MD,MO, MP,MT,MX conversions specified. | D,G,MD,MF,MP,MT,MX,T,U conversions specified. | D,ML,MR,MT,MX,T conversions specified. |
| **IF statement.** | Available. | THEN and ELSE clauses may start a line. | Available. | THEN clause is optional if ELSE is present. Multiline ELSE clause starting on THEN line is not specified. |
| **INDEX function.** | Available. | Input cannot exceed 256 characters. | If substring expression is null then occurrence number is returned. | If substring expression is null then 1 is returned. |
| **INMAT function.** | Returns number of elements loaded after MATREAD, or modulo after OPEN. | Returns number of elements loaded after MATPARSE, MATREAD or MATREADU, or modulo after OPEN. Invalidated by a SELECT. Returns list of dimensions for an array name. | Not available. | Not available. |

|  | Cosmos | Prime | Microdata | ADDS |
|---|---|---|---|---|
| **INPUT statement.** | Available. | Negative length expression causes test for presence of input without actually causing input. Backarrow/underscore indicates truncation to specified length should occur after carriage return. Maximum input length is 188 characters. | Backarrow/underscore indicates carriage return is still required when length specification is included to limit input. Maximum input length is 140 characters. | Masked version allows input and validation at a given screen position. |
| **$INSERT statement** to insert source code at compile time. | Nesting allowed. Delimiter is a comma when file is specified. | Nesting not allowed. Treename required and delimiter is angle bracket when file is specified. | Not available. | Not available. |
| **INT function.** | Truncates toward minus infinity. | Truncation direction not specified. | Truncates toward zero. | Truncation direction not specified. |
| **LOCATE statement.** | USING clause determines scope, BY clause not available. | Starting position is combined with attribute expression. | Available. | THEN clause available. All parameters stored as a list in function notation. |
| **LOCK, UNLOCK statements.** | Not available. | Locks are not cleared at program termination. | Available. | Locks are numbered from 0 to 47. |
| **LOOP statement.** | DO is optional. | Do is optional. | Available. | Available. |
| **MATREAD statement.** | INMAT function can return number of elements read. File variable required. | Zero element of matrix set to overflow characters. INMAT function can return number of elements read. | Overflow is lost. | Available. |
| **MATWRITE statement.** | TO can be used instead of ON. File variable required. | TO can be used instead of ON. Zero element may be written as a trailing field. STATUS function can return lock state. | Available. | Available. |
| **MOD, REM functions.** | REM instead of MOD is allowed. | MOD available only. | MOD available only. | REM instead of MOD is allowed. |
| **ON GOSUB, ON GOTO statements.** | Label colons are optional. | One statement may be broken into multiple lines. Label colons are optional. | Available. | No action taken if index is out of range. |
| **OPEN statement.** | DICT expression and TO clause required. | DICT expression is required. STATUS function can return file type, INMAT function can return file modulo. | Available. | Available. |
| **PAGE statement.** | Not available. | ON clause allowed. | Available. | Optional expression can change page number. |
| **PRECISION statement.** | Not available. | From 0 to 10 digits. Controls precision when numbers converted to strings. May be adjusted during program execution. | From 0 to 6 digits. Controls precision of calculations. May not be changed once set. | From 0 to 4 digits. Controls precision of calculations. May not be changed once set. |
| **PRINT statement.** | Comma zones are 16 spaces. Logical and comparison expressions must be in parentheses. ON clause not available. | Comma zones are 10 spaces. | Comma zones are 18 spaces. | Comma zones are 18 spaces. |
| **PRINTER statements.** | CLOSE not available. | ON clause available for CLOSE. | Available. | Available. |

| | Cosmos | Prime | Microdata | ADDS |
|---|---|---|---|---|
| **PROCREAD, PROCWRITE statements.** | Not available. | Ignored if compiling with option A, else unavailable. | Available. | Available. |
| **PROMPT statement.** | Controls display of values from DATA lists. | Controls display of values from DATA lists. | Available. | Available. |
| **READC statement.** | Not available. | Available. | Variable is set to status byte on error. | Not available. |
| **READNEXT statement.** | Value pointer not available. | FROM clause specifies list number. Optional subvalue parameter can return subvalue pointer along with value pointer for exploded lists. | Available. | FROM clause specifies list variable. |
| **READT, WRITET, WEOF, REWIND statements.** | Not available. | UNIT clause specifies unit number and I/O parameters. STATUS function can return tape status. | Available. | Available. |
| **READU, READVU, RELEASE, MATREADU statements.** | Not available. | STATUS function can return number of user that caused LOCKED clause to be taken. | Available. | LOCKED clause not available. |
| **REMOVE function and statement** to delete dynamic array substrings. | Statement form only. Required AT clause specifies starting position. | Available. | Not available. | Not available. |
| **RETURN statement.** | Optional before the final END in an external subroutine. | Optional before the final END in an external subroutine. | Available. | Available. |
| **RND function.** | Available. | Generator can be initialized to repeat sequence. | Argument must not be negative. | Argument must not be negative. |
| **RQM statement.** | Not available. | Terminates timeslice. | Sleeps one second. | Terminates timeslice. Optional sleep parameter allowed. SLEEP can be used instead of RQM. |
| **SELECT statement.** | Not available. | Invalidates INMAT value. Can assign list to a number from 1 to 10. | Available. | Can select from a file variable or attributes from a string. Can assign list to a variable. |
| **STATUS function.** | Available, but not fully documented. | Returns status of last FMT, ICONV, MATWRITE, OCONV, OPEN, READT, REWIND, WEOF, WRITE, WRITET or WRITEV operation. | Not available. | Not available. |
| **STOP statement.** | Optional expression is output at terminal. | Optional expression is output at terminal. | Optional expression list is used to output ERRMSG items. | Optional expression list is used to output ERRMSG items. |
| **SUBROUTINE statement.** | MAT arguments not specified. | Argument list may be broken across multiple lines. | Available. | Available. |
| **SUM function** to sum all values at lowest level of a dynamic array. | Available. | Available. | Not available. | Not available. |
| **TRIMB, TRIMF functions** to trim blanks at back or front. | Available. | Available. | Not available. | Not available. |

| | Cosmos | Prime | Microdata | ADDS |
|---|---|---|---|---|
| **WRITE, WRITEV statements.** | TO can be used instead of ON. File variable required. | TO can be used instead of ON. STATUS function can return lock state. | Available. | A zero or negative attribute number causes insertions at the beginning or end of items. |
| **WRITEU, WRITEVU, MATWRITEU statements** for writing without unlocking groups. | Not available. | Available. | Not available. | Available. |
| **Assignment operators.** | + =, − = and := available. | + =, − = and := available. | Not available. | Not available. |
| **Concatenation precedence.** | Lowest. | Lowest. | Highest. | Highest. |
| **Exponentiation operator.** | ** or ^ . | ** or ^ . | Not available. | Not available. |
| **Format specifications** for PRINT and assignment statements. | Format string is the same as provided by the FMT function. | Available only if compiling with option A. Formats are not specified in the documentation, but are probably the same as for the FMT function. | Format string is: justification (optional L or R), conversion (optional $, comma, number of fractional digits), field size (# list or # and integer). | Format string is: justification (optional L or R), conversion (optional number of fractional digits, scaling factor, zero suppression, comma, credit indicator, $), mask (optional #, * or %). |
| **Item identifier maximum length.** | 50 characters. | 188 characters. | 50 characters. | Not specified. |
| **Labels on statements.** | Alphanumeric labels allowed, must be followed by colon (colon optional on numeric labels). | Alphanumeric labels allowed, must be followed by colon (colon optional on numeric labels). | Numeric only. | Numeric only. |
| **Numbers** | Up to 18 digits in constants. E exponent notation allowed. $\pm 10^{4932}$ calculation range. | Up to 15 digits in constants. E exponent notation allowed. $-2^{47}$ to $2^{47}-1$ calculation range. Embedded blanks are ignored in arithmetic operations. | Up to 15 digits in constants. ± 140737488350000 calculation range. | Up to 15 digits in constants. ± 140737488350000 calculation range. |
| **Pattern matching.** | String expression and pattern maximum length is 254 characters. | Can use "..." for "0X". String expression must be less than 2,000 characters. Pattern must be less than 188 characters. | Available. | Available. |
| **Predefined @ variables** for access to system parameters. | 18 names. | 15 names. | Not available. | Not available. |
| **Relational operators.** | #>, #<, = > and = < allowed. | #>, #<, = > and = < allowed. | < > allowed. | < >, = > and = < allowed. |
| **Strings.** | Up to 254 characters in constants. Up to 65,530 characters in memory. | Up to 255 characters in constants. Up to three million characters in memory. | Up to 32,267 characters. | Up to 32,267 characters. |
| **Substring and field assignment** with bracket notation on left side of assignment statement. | Available. | Available. | Not available. | Not available. |
| **Variable identifiers.** | Alphabetic characters must be capitals. Maximum length not specified. | Maximum length is 50 characters. | Maximum length is 32,267 characters. | Maximum length is 64 characters. |

| | Cosmos | Prime | Microdata | ADDS |
|---|---|---|---|---|
| **@ function.** | Available. | Two zero arguments turn off pagination. | Negative arguments not allowed. | Available. |
| **[ ] Substring extraction.** | Negative starting position is used as a count from the end of the string. Negative length returns reversed string. Length expressions starting with F or B cause forward or backward search, and COL2 function can then return delimiter position. | Length expression specifies characters from end of string if starting position is omitted. | Available. | Available. |
| **Unique features** for each implementation. | • INP, OUT statements input and output values via ports. • File variables are required in READ, READV, DELETE statements. • BITAND, BITNOT, BITOR, BITXOR functions perform Boolean arithmetic. • CALCULATE, { } functions invoke dictionary word definitions. • DRIVE function reports default disk drive. • FIELDSTORE function replaces, deletes or inserts substrings. • QUOTE function surrounds expressions with quote marks. • XLATE function retrieves fields from a file. • ASCII, EBCDIC functions are not available. • CHAIN, ENTER statements are not available. | • An expression can be a conditional, allowing statements like PRINT SPACE (IF WIDE THEN 10 ELSE 1). • Indirect CALLs change the indirect variable to allow faster subsequent CALLs. Indirection is not necessary if subroutine names begin with ! or *. • CALLs to eight library subroutines to provide certain system parameters and functions are allowed. • MATPARSE statement maps delimited substrings into matrices. • MATCHFIELD function extracts substrings that match a pattern. • REUSE function pads dynamic arrays to provide equals lengths for arithmetic and other list operations. REUSE examples imply that arithmetic operators such as + will work with a list of numbers, but this is not documented. • IF $TRUE/$FALSE statements provide conditional compilation. • 22 library subroutines provide operations on two dynamic arrays, resulting in one dynamic array. • ITYPE function allows access to dictionary definitions in files. • READV/READVU statements with attribute zero for Type 1 files test for existence of an item without actually reading. • OPENSEQ, READSEQ, WRITESEQ, WEOFSEQ, CLOSESEQ statements provide sequential file I/O. • FILELOCK, FILEUNLOCK statements lock and unlock every item in a file. • COMMON and DATA statements may be broken into multiple lines. • Labeled COMMON is allowed. • TIME function returns hundredths of a second. | • SHARE statement shares one catalogued copy of constant data with other processes. • INPUT and MATINPUT statements with USING clause perform input under control of separately compiled screen processor definitions. • Type ahead is not available. | • ALPHA function validates alphabetic strings. • CRT statement always PRINTs to display screen. • DCOUNT function counts number of values separated by a delimiter. • INPUTERR statement displays error messages. • INPUTTRAP statement defines program branches to be taken upon detection of certain input characters. • INPUTNULL statement defines input equivalent to null. P |

## Some New Subscribers

**Ed Sheehan**
Certified Appliance Distributors
Los Angeles, CA

**Kevin J. Cook**
ADP
Roseland, NJ

**Richard Chess**
General Piping Products
Houston, TX

**Richard B. Usry**
Minicomputer Co.
Richmond, VA

**Brian C. Ehlers**
Dominguez Water Corp.
Long Beach, CA

**W. Shave**
Cavalier Carpets of New Zealand Ltd.
Papatoetoe, Auckland

# An Undocumented Editor Capability

**The editor's "not found" modifier for testing the success or failure of a previous L command is explained.**

A capability in the Microdata™ editor (and possibly other Pick-based editors) for testing the success or failure of a previous locate (L) command is generally unknown among users because for some reason it has remained undocumented. By prefixing an editor command with a semi-colon (;), the command will be executed only if the previous locate failed (in other words, the string last searched for was not found). Here is an example of using this "not found" test:

```
:SELECT PROGRAMS
3  ITEMS SELECTED.
:EDIT PROGRAMS
PROG1
TOP
.PL/IF ERR42 THEN STOP[;P1[DE[FI[P
.P1EX[P
```

In this example, a complete file of program items is being edited. In each program, the user wants to delete the first line containing the string "IF ERR42 THEN STOP", but only those lines. The first editor command shown above uses the Prestore facility to define a command string that (1) tries to locate the desired string, (2) executes the P1 prestored command if the string is not located, otherwise (3) deletes the line, (4) files the edited item and (5) re-executes the prestored command string. The second editor command defines the P1 command string as (1) an EXit (to avoid the DE and FI) followed by (2) a P to re-execute the first prestored command string. When used in this way, the two prestores essentially provide an IF-THEN-ELSE capability. Once the two command strings have been prestored, the user only has to type P to begin automatically processing every item in the file, knowing that the ";" test will make sure that only items containing the string in question will be edited and filed. If the ";P1" clause were not included, the search would fail in any item not containing the string, and DE would delete some arbitrary line. What would happen if only the first prestored command were used with ";EX" in place of ";P1"?

Thanks go to Ed Sheehan of Certified Appliance Distributors for pointing out this undocumented feature. Ed mentions it might also apply after an unsuccessful replace (R) command, but only thorough testing can verify that. The example above was tried on a 3.2B Reality®. It has not been tried on a Sequel™. On what commands the ";" prefix will work, and how soon after a search does the test have to be made, remain unanswered questions. Note that since features such as this are usually undocumented because of bugs or because of no plans for future support, they should only be used as a last resort. Such features should never be left permanently imbedded in a piece of software such as a proc.

# local user groups

**Delaware Valley**
The first meeting of the newly re-organized Delaware Valley Data Base Management Association was held in March. At that meeting, Bob Clearfield of Delaware Valley Underwriters reported on the group's software library and reviewed the Phoenix MICRU conference. Jim Reilly of Microdata gave a presentation describing the 4.2 release. Dues are $25 per company for one member, plus $5 per additional member. The group is in the process of expanding to include other Pick users, not just Microdata sites. Prospective members should contact President Jim Cates, Pars Manufacturing, Box 149, Ambler, PA 19002.

**Arizona**
The Phoenix/Tucson Pick user group started just over one year ago and now has about 20 active members. The group exchanges ideas and, after a 10 to 15 minute business meeting, sponsors educational sessions at each gathering, so far having covered ENGLISH®, the editor and proc. Shelly Frecker of Alphagraphics in Tucson is the elected chairperson. Dues are $25 for the first person from an organization and $15 for each additional member. Potential new members should contact Secretary-Treasurer Jodi Hilgenberg, Communication Skill Builders, Box 42050, Tucson, AZ 85733.

**Northern California**
The Northern California Pick Users is a group that meets once a month, each time at a different San Francisco Bay area restaurant. Board meetings are at 5, cocktails at 6, dinner at 7, and a presentation at 8:30. The February meeting featured a visit by Mark Pick and Mike Sibley of General Automation to discuss their new Zebra machine, March offered a panel discussion by dealers for Prime, Ultimate, ADDS and Microdata, and April's presentation was titled "Communications Concepts for Beginners", by Henry Herman of SKP Electronics. Annual dues are $30 for a User Member or $45 for an Associate Member, and dinners are $20 for members, $25 for non-members. Interested users should contact Secretary-Treasurer Lisa Levsen, Cornnuts, Box 6759, Oakland, CA 94603.

**Why isn't YOUR user group mentioned above? To keep us informed of your group's activities, and to appear in future issues of Pragma, keep us on your mailing list! Send your newsletter, announcements and press releases to: Pragma, 207 Granada Drive, Aptos, CA 95003.**

**Command files are typically the "glue" that holds the components of a software system together. Microdata's command file capability is called PROC, Prime offers Perform, and so on. In this regular department, Pragma will be presenting concepts and techniques to help installations squeeze the most out of their command file processors.**

# Avoiding Saved Lists with PQ-RESELECT

**The PQ-RESELECT verb can be used to avoid the slower SAVE-LIST, GET-LIST, DELETE-LIST method of file processing.**

The two procs on the right are both designed to do exactly the same thing: compile and catalog all source code items in a file on a Microdata$^{TM}$ Reality® system. The first proc uses a traditional and often seen method that SELECTs the appropriate items out of the file and first saves the list (here under the name of PROGRAMS) instead of immediately compiling the items. By saving the list, a fast GET-LIST can then be used to start the subsequent CATALOGing loop, instead of having to slowly re-SELECT all of the items again. Note that at the end of the proc, a DELETE-LIST is used to get rid of the temporary PROGRAMS list.

The first proc works fine, but consider the second version shown below it. This second version does just as good a job of compiling and cataloging all SELECTed items in a file, but in a much simpler and quicker fashion. By using the PQ-RESELECT verb to re-initialize the SELECTed list "register", all SAVE-LIST, GET-LIST and DELETE-LIST statements are avoided, shortening and simplifying the proc and speeding it up even more. Also note that if the operator happens to abort this proc by using the BREAK key while it executes, no intermediate list will be left over, as would happen with the first proc.

These examples show how easily PQ-RESELECT can be used to improve proc size and performance. The technique can be adapted to just about any application that needs to process a SELECT list multiple times. Ⓟ

```
SLOW.INSTALL
PQN
HSELECT BP # '$]' AND # '*]'
STON
HSAVE-LIST PROGRAMS
P
HGET-LIST PROGRAMS
STON
HPQ-SELECT 1
P
100 MV %1 !1
IF # %1 GO 200
HBASIC BP
A
P
GO 100
200 HGET-LIST PROGRAMS
STON
HP@-SELECT 1
P
300 MV %1 !1
IF # %1 GO 400
HCATALOG BP
A
P
GO 300
400 HDELETE-LIST PROGRAMS
P
```

```
FAST.INSTALL
PQN
HSELECT BP # '$]' AND # '*]'
STON
HPQ-SELECT 1
P
100 MV %1 !1
IF # %1 GO 200
HBASIC BP
A
P
GO 100
200 HPQ-RESELECT 1
P
300 MV %1 !1
IF # %1 X
HCATALOG BP
A
P
GO 300
```

Ⓟ

# Self-Documenting Reports

**A small proc and program for automatically documenting reports are presented. The programs cause command lines to be imbedded in the headings of output reports.**

In Pragma's February issue (Pragma #3, page 27), Wish List item #54 stated: *If an ENGLISH® command does not include a HEADING clause, then a default heading that shows only the page number, date and time is generated for the report. It would be much more useful for the default heading to also include the text of the command itself that generated the report. In this way, ENGLISH reports become self-documenting. A glance at the header on any report page will tell the reader what command can be used to duplicate the report.* The following simple Reality® proc and program will automatically provide much of the capability that the Wish asks for:

```
HEADUP
PQ
HRUN BP HEADUP
P

HEADUP
PROCREAD COMMAND ELSE STOP
BLANK.POS = INDEX(COMMAND," ",1)
IF BLANK.POS = 0 THEN STOP
COMMAND = COMMAND[BLANK.POS+1,LEN(COMMAND)-BLANK.POS]
COMMAND = COMMAND:' HEADING "':"PAGE 'P' - ":COMMAND:
" - 'TL'":'"'
CHAIN COMMAND
END
```

Once the HEADUP proc has been installed in a Master Dictionary, and the corresponding HEADUP program is in place in the BP file, a "verb" named HEADUP effectively becomes available to the user. By simply prefixing most ENGLISH commands with the word "HEADUP", the same report the command normally generates will be output, but the report heading will now show the command that caused the report, thereby documenting how the report was created. For example, the command LIST MD 'HEADUP' will output the MD proc definition in report form, but the heading on the report will only show the page number, date and time. By using the prefixed command HEADUP LIST MD 'HEADUP' instead, the same report will be output, but the heading on the report will additionally show the LIST command that was entered. Unfortunately, the syntax of the HEADING clause handles some single quoted characters in a special way (such as 'T' causing the time and date to be inserted), so the HEADUP program will not work properly with some commands that happen to use such quoted characters in item lists (although the above example with single quotes happens to work). Also, command lines with double quotes, such as those in value tests, can never be prefixed with HEADUP, since the HEADUP program then tries to imbed the command line inside its own HEADING clause delimited with double quotes, resulting in an invalid use of double quotes.

Note that since HEADUP effectively appends a HEADING clause to the command being used, a command prefixed by HEADUP that already has a HEADING clause will override the HEADUP-generated HEADING (since the system ignores all but the first HEADING clause), thereby canceling the effect and preventing the report from being self-documenting. Also, users who attempt to "completely" document their reports by using two HEADUPs at the start of a command will be surprised to discover that the system aborts! But even with these limitations, HEADUP can be a useful tool for easily and quickly documenting many ENGLISH reports in a convenient manner.

# Queries

**If you have questions or if you have answers, send either to Pragma for publication — both are eligible for Pragma's author payment awards.**

*The previous 23 Queries have been featured in issues #1 through #3 of Pragma. Seven of those Queries are still unanswered.*

**24. Our machine is getting bogged down, and we can't afford to buy more hardware. What are some quick fixes we can try to speed up the system?**

Try doing the following on a regular basis, as a form of software preventive maintenance, in order to keep your system performance as high as possible:

*A. Periodically re-allocate file modulos.* Use techniques like that shown on page 6 of Pragma #2 to set your file modulos. Do a file restore about once a week or whenever your files have grown substantially, in order to re-allocate the files, compact the disk, and recover "lost" frames.

*B. Monitor large files.* Sort the STAT-FILE by descending frames to determine what files are largest. Delete obsolete, unused or useless files and file items.

*C. Catalog all programs.* Don't use the RUN verb. Insure that users never execute separate copies of a program. Only one cataloged version of any program should exist, and all users should execute the one version via the verb created in one account by the CATALOG command.

*D. Monitor CPU consumption.* Use the history information recorded by the system in the ACC file to determine which accounts and users are consuming the most CPU time. Don't waste time trying to improve just any applications program or file — only worry about optimizing those programs and applications that consume the most CPU time or perform the most disk reads, as indicated by the statistics in the ACC file.

*E. Avoid SAVE-LIST.* Review all applications that do repeated SORTs or SELECTs. Use PQ-RESELECT instead of SAVE-LIST, as shown on page 26 of this issue.

*F. Share accounts.* Whenever possible, configure accounts so that they all share one Master Dictionary, instead of having one MD for each account. Remove all unused and unnecessary items from every MD, and be sure to set the correct modulo for each MD, just like any other file.

*G. Compile all procs that use GO commands.* This typically includes all menu procs. Use the PQ-COMPILE verb.

*H. Place all procs in one file.* A proc library can be created and shared by all accounts. There should never be more than one copy of a proc on the system.

*I. Keep a small timeslice set on data entry accounts.*

*J. Compile with the option to leave end of line bytes out of object code.*

## A Survey of Hardware That Supports Pick-Style Software

| Max / Min | RAM Memory | Disk Memory | Backup Device | Serial Plus Parallel Ports | Purchase Price | Monthly Maintenance | Systems Installed | Contact |
|---|---|---|---|---|---|---|---|---|
| ADDS | 1 MB / 128 KB | 600 MB / 30 MB | ½" Tape / ¼" Tape | 64 + 1 / 8 + 1 | 150,000 / 25,000 | 1,300 / 280 | 600 | Robin White<br>ADDS Inc<br>100 Marcus Blvd<br>Hauppauge NY 11788 |
| Altos | 1 MB / 512 KB | 42 MB / 10 MB | ¼" Tape / ¼" Tape | 10 + 0 / 6 + 0 | 20,480 / 8,990 | TRW / TRW | 12 | Jim Renalds<br>Altos Computer Inc<br>2641 Orchard Pkwy<br>San Jose CA 95134 |
| ATV | 1 MB / 256 KB | 1.6 GB / 33 MB | ½" Tape / ½" Tape | 64 + 4 / 16 + 1 | 325,050 / 39,950 | 4,209 / 499 | 450 | Terry Mulhern<br>ATV/Evolution<br>2921 S Daimler<br>Santa Ana CA 92711 |
| CDI | 1 MB / 96 KB | 2 GB / 14 MB | ½" Tape / Floppy | 64 + 4 / 4 + 1 | 101,360 / 41,980 | 491 / 189 | 30 | Dennis Brown<br>CDI<br>1309 114th SE #300<br>Seattle WA 98004 |
| Cosmos | 576 KB / 320 KB | 12 MB / 160 KB | ¼" Tape / Floppy | 1 + 1 / 1 + 0 | 9,079 / 4,144 | 90 / 40 | 19 | Roger Harpel<br>Cosmos Inc<br>123 Ferntree Dr W<br>Morton WA 98356 |
| Datamedia | 1.5 MB / 256 KB | 124 MB / 12 MB | ¼" Tape / ¼" Tape | 16 + 1 / 1 + 1 | 46,530 / 16,740 | 557 / 257 | 6 | Paul Dahill<br>Datamedia Corp<br>7401 Central Hwy<br>Pennsauken NJ 08109 |
| General Automation | 1 MB / 256 KB | 632 MB / 20 MB | ½" Tape / ¼" Tape | 32 + 0 / 2 + 0 | 95,700 / 15,950 | 935 / 175 | 12 | Shirley Stough<br>General Automation<br>1055 S E St<br>Anaheim CA 92805 |
| Microdata | 4 MB / 64 KB | 1 GB / 32 MB | ½" Tape / ½" Tape | 127 + 4 / 8 + 1 | 389,185 / 26,985 | 3,761 / 345 | 7,200 | Linda Spelman<br>Microdata Corp<br>17481 Red Hill Ave<br>Irvine CA 92714 |
| Prime | 8 MB / 512 KB | 4 GB / 64 MB | ½" Tape / Disk | 128 + 4 / 8 + 1 | 356,000 / 39,900 | 2,980 / 380 | 400 | Lee Adamson<br>Prime Computer Inc<br>Prime Park<br>Natick MA 01760 |
| SMI | 4 MB / 128 KB | 4 GB / 28 MB | ½" Tape / ½" Tape | 128 + 2 / 4 + 0 | 800,000 / 45,025 | 5,200 / 530 | 10 | Jeanine Christiano<br>SMI<br>6300 N River Rd<br>Rosemont IL 60018 |
| Ultimate | 2 MB / 128 KB | 1 GB / 15 MB | ½" Tape / ¼" Tape | No practical hardware limit / 4 + 1 | 530,000 / 20,000 | 3,059 / 245 | 1,150 | Dick Gould<br>Ultimate Corp<br>77 Brant Ave<br>Clark NJ 07066 |

# A Survey of Hardware that Supports Pick-Style Software

**A table is presented showing the minimum and maximum hardware configurations and capacities for systems that support Pick-style software.**

The table on the opposite page shows the remarkable variety of hardware now available from U.S. vendors for Pick-style software. The following notes should be used in conjunction with the table:

Backup Device: Many vendors offer either tape or removable disk for the backup device. When there was a choice, vendors were instructed to choose the least expensive for the minimum configuration and the one with the greatest capacity for the maximum configuration. Note that even when two systems use the same sized backup media, there can still be a difference between systems in density, speed and capacity.

Purchase Price: Vendors were instructed to include hardware and standard software costs, and any applicable fees such as royalties. Vendors were told that the maximum configuration price was to include only the minimum number of printers, terminals and tapes, if any, that must be purchased.

Monthly Maintenance: Altos deferred to the TRW service organization for prices.

Systems Installed: Vendors were asked for the total number of Pick-style systems installed at end user sites (not including dealers, but including beta sites for pre-production machines) as of March 15, 1983. Vendors developing or planning systems, but who did not have any units installed, were not included in the survey.

Readers should note that Cosmos independently sells an unbundled Pick-style operating system for the IBM hardware quoted in their column. CDI and SMI also are not hardware manufacturers, but do sell hardware. Readers should also understand that even though a vendor claims to support a particular hardware model or configuration, it cannot be assumed that every model or configuration has been sold and delivered, or even engineered. Beware that some vendors tend to exaggerate certain figures, especially quantities shipped. Finally, remember that the number of machines installed is generally larger than the number currently operating, since a variety of forms of attrition (such as converting to another brand of hardware) tend to constantly erode a vendor's installed customer base. P

## the computer room

# Printer Trade-Offs

**The pros and cons of where to plug in a printer are described for the most popular system configurations.**

Many installations go on using their particular configuration of printers and terminals day after day, never considering whether a different hardware arrangement may be better suited to their needs. Here are the pros and cons of some of the different ways in which one machine might be arranged to print hard copy:

1. **Centralized system printers.** This is usually a high speed line printer interfaced to the system's primary parallel port. An advantage of this configuration is that such interfaces and printers can support high output rates, so that large listings are finished quickly. One printer can easily support multiple users, who simply have to add a (P) to their command to print their reports. Disadvantages are that such printers are typically in the computer room far away from users, that large numbers of reports may be queued up and have to wait for one another, and that it can be a nuisance to set up and change special forms on a printer shared among various jobs.

2. **Serial printers slaved off CRTs.** Advantages are that such printers are conveniently close to the CRT creating the report, they don't require an extra port, they don't have to be shared with other processes, and they are easy to dedicate to certain jobs or forms. Disadvantages are that the devices are generally fairly slow at printing, often require the host CRT to run at the same baud rate, and keep the CRT tied up while jobs are printing.

3. **Serial printers spooled via their own port.** This has some of the same advantages as method 1 above, in that such configurations can easily be shared, although users and programs will typically have to do a bit of set-up with SP-ASSIGN commands. Unlike method 2, this technique allows the printer to be nearby while leaving the CRT available for other work (at any baud rate), even when the printer is busy printing a report. The main disadvantage is that a port must be dedicated to the printer.

Note that all of the above configurations can be supported and mixed together on just one system. Also, there are now many devices on the market that can be inserted into the transmission line between printer and host (such as switches to share multiple printers on one port, or buffers to receive all data to free the host and then keep the printer busy at a slower rate) in order to improve performance and convenience. P

# An Introduction to ENGLISH® Part 3: Finding Files

**The third in a series of articles is presented for the beginning user of the inquiry and report generation language called ENGLISH. The Master Dictionary is introduced and the command for finding the files defined in the Master Dictionary is given.**

If you have carefully read the previous two installments, and have actually logged on to a terminal and tried to input the sample ENGLISH commands suggested in those articles in order to list files, and if you have memorized all the necessary jargon, then you should now know the following:

• The LIST ACC command lists the contents of a file named ACC which contains information describing how much each account is used on the computer. (How much information is stored in the ACC file is usually controlled by the same person who controls the passwords for accounts. Some installations choose to keep lots of data in the ACC file. Other installations choose, or get the default because no choice is made, for little or no data in ACC.)

• Using SORT instead of LIST causes output to be sorted by the item identifier. The COUNT verb only outputs the number of items in the file. The ONLY clause suppresses all output columns except for the item identifiers.

• The DICT clause causes the file's dictionary to be listed instead of the file's data. The dictionary contains words that can be used in ENGLISH commands to specify which columns of data should be output. For example, if A1 and A2 are words in the DICT of the ACC file, then LIST ACC A1 is a valid command and will list one column of data alongside the item identifiers. LIST ACC A1 A2 will include one more column on the report.

We know how to list and sort a file, and we can even list the available dictionary words for a file so that we can select which words to include in our command (since those words control which columns of data appear on the report that is output). But how do we know what files are available to list, other than the ACC file?

One file accessible to every account is called the Master Dictionary, or MD for short. It is an important file because it contains data that describes just about everything that can be done on the system. Included in the MD file are the names of all other files that you can access. By listing the MD file, you can get a list of the other files that exist. Use the command SORT ONLY MD WITH *A1 "D]" "Q" to get a list of available file names. Now that you know the names of other files, try to LIST each of those files in turn, to see if you can determine what is in each file. Remember to LIST the DICT of a file if you need to know what words can be used to cause more data columns to appear in the output. Try logging on to other accounts to see what files are available there. What happens when you input just SORT ONLY MD? P

# letters

**If you use a Pick system, Pragma wants to hear from you. Have you developed applications of interest to other users? Do you plan to acquire new hardware? What features would you like to see in Pragma? Are you active in any type of user group organization? Write Pragma today. All letters to the Editors are welcome, and as many as possible will be published in the Letters Department in every issue.**

## Another Edit Aide

I am writing in response to Mike Rossetti's article on *Edit Aides* (Pragma #3, page 6). When we started programming our machine some six years ago, Howard Marks gave us our first "BEC" — Basic, Edit, Compile. The concept of the program then was very similar to the one described in Mike's article. Since then, we have gradually refined the proc and added additional capabilities and now, of course, find it something we could never live without. Since it seems to have a number of significant enhancements, I thought you and perhaps some of your readers would be interested in knowing about it. We don't claim that it is unique, for on the contrary, it was born out of necessity and simply utilizes system capabilities and programs that have appeared elsewhere, but putting them altogether in one proc makes for a pretty powerful combination.

Jack Hardman
and the programming staff
Hardman Inc.
600 Courtland
Belleville NJ 07109

*We have talked with Jack and he has agreed to make the software available to anyone who sends him a self-addressed stamped magnetic tape. Recipients of this set of programs will discover they have acquired a 230 line proc supporting 15 different command options such as edit, compile, run, renumber (we're flattered!) and "nice": a 335 line program by Brian and Mark Hill to change the old EXTRACT and REPLACE parenthetic forms into the nicer syntax using angle brackets. Nice also changes occurrences of the SPACE function into format conversions, and breaks up long lines into short lines for reading convenience on an 80 column screen.*

*—The Editors* P

# Which PICK™-based electronic spreadsheet should you choose?

## COMPU-SHEET.™

**You already know the value** of an electronic spreadsheet. The question is which one should you choose? The answer is easy. COMPU-SHEET.

**In the past six months,** over 150 end users have chosen COMPU-SHEET for their PICK-based computer systems. And more importantly, after months of extensive evaluation and comparison, CDI, General Automation and Datamedia have selected COMPU-SHEET as the best electronic spreadsheet for their customers. Those are pretty powerful endorsements, and the list is growing every day.

**Why have so many** people chosen COMPU-SHEET? Just compare COMPU-SHEET's key features with any other spreadsheet program:

- Very easy to use...no programming experience is needed.
- Uses simple English-like commands (COPY, MERGE, PRINT...)
- Worksheets may be merged or consolidated. The user controls the range of locations to be merged or consolidated.
- Formulas may specify retrieval of data from other worksheets.
- Formulas may specify retrieval of data from any file in the user's data base (down to the *sub-value level,* if desired).
- The user may enter a "?" at any point to get a "HELP" display.
- Multiple windows provide for viewing various sections of the worksheet at one time.
- The user has the option to control the format of any printed reports. Any combination of columns and rows may be selected.
- Formulas are written in simple algebraic format. Even "IF condition THEN formula1 ELSE formula2" is supported.
- A complete and formatted "AUDIT" report is available.
- Interfaces with "ACCU-PLOT" graphics system. Your worksheets may be represented as line graphs, pie charts, bar graphs, etc.
- Operates on Microdata, all PICK systems, and the Prime Information System.
- And more...

**Plus,** Interactive Systems offers you complete support after you choose COMPU-SHEET. If you ever have a question, simply call us at (602) 993-3579, and we'll get you a helpful, friendly answer *fast.*

**We want you to decide** for yourself. Ask about our no-risk 21-day trial period. For information and a complete brochure, call Interactive Systems today or mail the form below. We're sure you'll choose COMPU-SHEET too.

**INTERACTIVE SYSTEMS**
129 East Voltaire Avenue
Phoenix, Arizona 85022
(602) 993-3579

**Yes! Please send me more information about COMPU-SHEET.**

Name ____________________

Company ____________________

Address ____________________

City ____________________

State ____________ Zip ________

Phone (    ) ____________________

COMPU-SHEET is a trademark of Interactive Systems. PICK and PICK Systems are trademarks of PICK Systems, Irvine, CA.

# VANILLA (The No-Frills Manufacturing System) Part 4: Purchase Order Entry

**The fourth in a series of articles on the design and implementation of Vanilla, a software system for manufacturing, is presented. Software and techniques for the creation and maintenance of a purchase order file are described.**

The previous installment in this series (Pragma #3, page 30) described some of the attributes necessary in a purchase order file. Shown below is the actual SYSMAP-style file format documentation for Vanilla's purchase order and vendor master files. The A/V/S column denotes whether data is stored as an attribute, value or subvalue (see the SYSMAP articles in this and prior issues of Pragma for more thorough descriptions of how SYSMAP documentation is organized).

The listing beginning on page 36 is the complete Vanilla program, named GET.PO, for maintaining the purchase order file. Although the program is one of the largest in the Vanilla repertoire, that is mostly because GET.PO is designed to be a flexible, easy to use purchase order entry program. Also, GET.PO can be easily adapted to handle purchase orders for a wide variety of application environments, not just manufacturing. GET.PO allows the operator to create purchase orders

```
File.. Attribute...... AMC * Description.................................

PO                             Purchase order file.
PO     NUM               0     Purchase order number.
PO     VENDOR            1 A   Vendor number for this purchase order.
PO     DATE              2 A   Purchase order date.
PO     NOTES             3 V   Notes about this purchase order (not tied to
                               just one item).
PO     TERMS             4 A   Terms of purchase.
PO     QA                5 A   "Y" if deliveries need to go through
                               receiving inspection else "N".
PO     PO.TOTAL          6 A   Total value of purchase order to two decimal
                               places.
PO     ITEM              7 V   Item numbers. At least the first ("1") is
                               required.  The numbers are always in order
                               with no gaps.  All attributes from PART on
                               down are associated fields.
PO     PART              8 V   Our part number, if any, being purchased.
PO     UM                9 V   Item's unit of measure.
PO     DATE.IN          10 S   Actual delivery dates, in accumulated order
                               of entry, that parts were received from the
                               vendor.
PO     SCHED            11 S   Delivery dates, in ascending order, for this
                               item as entered on purchase order. QTY,
                               BALANCE and UNIT.PRICE are associated fields.
PO     QTY              12 S   Quantity scheduled for delivery.
PO     BALANCE          13 S   Item quantity remaining open for each
                               delivery.
PO     UNIT.PRICE       14 S   Unit price per delivery to five decimal
                               places.
PO     DESC             15 V   Item description.
PO     VPART            16 V   Vendor's part number.
PO     ACCOUNT          17 V   General ledger account number for charging
                               the purchase.
PO     EST              18 V   Estimated unit cost of item to five decimal
                               places.
PO     NOTE             19 V   One line note to help describe item being
                               purchased.

***

VM                             Vendor master file.
VM     NUM               0     Vendor number.
VM     NAME              2 A   Vendor name.
VM     ADR1              3 A   Vendor address line 1.
VM     ADR2              4 A   Vendor address line 2.
VM     CITY              5 A   Vendor city.
VM     STATE             6 A   Vendor state.
VM     ZIP               7 A   Vendor ZIP code.
VM     PHONE             8 A   Vendor phone.
VM     CONTACT           9 A   Vendor contact name.
```

**GET.PO**
**FILE FORMATS**

℗

# MULTIDATA'S NEW FIXED ASSET MANAGEMENT SYSTEM

**...Handles the Exceptions as well as the Rules...**

Multidata Corporation offers the most comprehensive on-line systems available. Each system is ergonomically designed to be truly user-friendly. Comprehensive "help" messages are provided at every data-input location. User-definable formats for data entry and retrieval let the system conform to your current business structure.

The Fixed Asset Management System provides a total solution to this complex problem. A representative sample of the features are:

**CONTROLLER**

- Corporate Elections/Standards
- All Depreciation Methods & Conventions including ACRS
- Handles non-standard items, such as partial fiscal year, 13 Accounting periods, or 4-4-5
- Depreciation projections, by period and G/L account
- Over 30 reports, each having multiple selections

**TAX ACCOUNTANT**

- Book federal and multiple state calculations
- ITC Calculations
- Ability to amend prior year records
- Gain/(loss) calculations
- Reconciliation of book-to-tax
- Ability to "what if" depreciation methods on an asset or entire property group

**PROPERTY MANAGER**

- Asset description
  —Vendor/manufacturer
  —Serial/Model Number
  —Location/Assignment
  —Tag Number
- Asset acquisition/disposition
- User definable formats
- Multiple locations (state, county, city, building)

By using Multidata's full support program, you are protected against any future changes in the law obsoleting your system.

## ANOTHER MULTIDATA PRODUCT...

The Agency Management Support System provides Insurance Agencies and Brokers with the latest on-line data base technology for:

- Automated Case Tracking
- Marketing/Sales Support
- Electronic Filing/Mail
- Data Base Analysis
- Financial Analysis

Increase your profitability through greater productivity. A General Information Manual and Demonstrations are available. We would like the opportunity to tell you more about this Information Management System.

MDC MULTIDATA CORPORATION

**Integrated Solutions For Business**

**1570 The Alameda, Suite 110 • San Jose, California 95126 • (408) 293-1801**

**FOR SALE**

**Microdata™ REALITY®6000**

128KB MOS Memory
2 50MB Reflex® discs
3 Open ports
2 Prism® CRT terminals
300 LPM line printer

Entire system available or parts may be purchased.
Call Paula Mannillo at 612-292-0199.

**128MB REFLEX II**
**FOR SALE**

Write or call:

Kent Sulprisio
Zentec Corporation
2400 Walsh Avenue
Santa Clara, CA 95050
408-727-7662

**MICRODATA®**

USED EQUIPMENT, SYSTEMS

- MOS SYSTEMS
- CORE SYSTEMS
- 50 MB REFLEX I, DISC DRIVES
- 50 MB REFLEX I, SUBSYSTEMS
- 128 K MOS MEMORY BOARDS
- 16 K CORE MEMORY BOARDS
- PRISM I CRTS
- PRISM II CRTS
- ADDS VIEWPOINT CRTS
- 8-WAY BOARDS, WITH PORTS
- PRINTRONIX PRINTERS & CONTROLLERS
- ROYALE B EXPANSION CABINET WITH 1455 BACKPLANE, CHASSIS, AND POWER SUPPLY
- SOFTWARE

CALL CHUCK HAAS FOR COMPETITIVE PRICES

513- 528-5733

EASTCOMP II

for any number of line items. Each line item can have any number of deliveries. Deliveries are automatically maintained in due date order, and fields such as delivery balances and item totals are automatically computed and continuously displayed, so that their current values are always shown. Line items and deliveries can be added, deleted or changed, with GET.PO automatically checking receipt status to verify that all edits are valid. The GET.PO display shows all purchase order fields at once, and individual line items and deliveries can be easily selected for viewing.

Included here are only brief descriptions of the general features and capabilities of GET.PO most likely to be modified for a given installation. As additional Vanilla applications are presented (particularly receiving and inspection) and the purchasing loop is closed, there will be more discussion regarding the tradeoffs involved in designing a purchasing system. Note that in the following descriptions, [numbers in brackets] refer to line numbers in the GET.PO program listing.

• The vendor record is locked [395] during GET.PO execution, so adding attributes to the VM file, such as year-to-date purchase order counts or dollar totals, can easily be added. For example, if a new VM attribute is to track the dollar value of all purchase orders for a vendor, use a variable to save the starting total order price when the order is first retrieved [89] and then subtract that amount from the new order total when the order is finally filed [348]. Add the resulting delta amount to the new VM attribute and change the RELEASE to a WRITE [349] so that the VM file stays up to date. Note that as long as there have been no receipts, the vendor for a purchase order can be changed [296].

• Any number of purchase order fields stored as attributes (similar to the terms or QA fields) can be easily added to GET.PO by providing new input prompts along with the existing fields before the item input loop [117 to 118] or after [249]. Remember to also provide a way to change those new fields by using code similar to [300 to 301 or 325 to 326].

• Any number of line item fields stored as values (similar to the unit of measure or account fields) can be easily added to GET.PO by providing new input prompts along with the existing line item fields [146 to 157]. Remember to also provide a way to change those new fields by using code similar to [308 to 313], to display those fields whenever the item is refreshed on the screen [406], and to delete those fields if the item should be deleted [418].

• The operator is allowed to exit prompts at will to get to the final file-or-edit prompt [261]. This means that required fields can be skipped during the creation of a new purchase order, so all required fields must be tested for the presence of data [342 to 345] to make sure the order is not filed with an empty required field.

Some of the purchase order attributes supported by GET.PO, such as the terms field or the total order amount, are technically of no interest to a software system mainly designed to perform MRP. However, a few such fields generally must be included to keep the program practical enough to be useful for purchasing, so they have been included in Vanilla. Note that the QA and line item account number fields are important to Vanilla because the account number will be used to indicate what items are being bought to inventory (as opposed to office supplies and other such purchases), while the QA flag will control whether inventory quantities will get detoured through receiving inspection. In the next installment, receiving and its interface with the purchasing system will be the subject for discussion. P

NEW

## JUST RELEASED: THE PICK Pocket Guide™

From the creators of the user friendly PICK data base management operating system, the new industry standard, a handy, single-source reference to its:

- VERBS
- SYNTAX REQUIREMENTS
- COMMAND FORMATS
- AVAILABLE OPTIONS

Used with all PICK operating systems, regardless of hardware manufacturer or system vendor, the PICK POCKET GUIDE is conveniently small, easily used, and indexed to quick reference sections that include:

- Editor • ACCESS • PICK/BASIC • Spooler
- Runoff/JET • ASCII Tables • PROC
- Debugger • Error Messages • TCL

**PERIODIC UPDATES:** For a small annual maintenance fee, PICK SYSTEMS™ will provide timely up-dates, including publisher enhancements and all system changes.
The PICK Pocket Guide will increase efficiency and help maximize system capabilities. Try it.
Act now. Order the PICK Pocket Guide from your dealer, or from PICK SYSTEMS with the order form below.

Actual Size

**Order Form**

PICK T.M.
SYSTEMS

17911-D Skypark Circle
Irvine, CA 92714
(714) 540-4048

**Name:** ________ **Company Name:** ________
**Address:** ________ **Telephone:** ( ) ________
**City:** ________ **State:** ________ **Zip:** ________

Yes, send                                                    Sub-total ____
___ PICK Pocket Guides @ $50.00 each ___ Total     †Shipping ____
___ Total Update Service @ $10.00 ___ Total     *Tax ____
*California Residents add 6% sales tax.
†Shipping charges: U.S. $2.50 per order, outside U.S. $7.50 per book.     Total ____
☐ Check or Money Order enclosed for $____     ☐ Mastercharge ☐ VISA
☐ Charge my credit card account for $____     Expiration Date: ___/___
Credit Card No.: ☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐     PAD
Signature: ________

## GET.PO PROGRAM LISTING

```
EQU PM$DESC TO 2
EQU PM$UM TO 3
*
EQU PO$VM.ID TO 1
EQU PO$DATE TO 2
EQU PO$NOTES TO 3
EQU PO$TERMS TO 4
EQU PO$QA TO 5
EQU PO$TOTAL TO 6
EQU PO$ITEM TO 7
EQU PO$PART TO 8
EQU PO$UM TO 9
EQU PO$RECEIVED TO 10
EQU PO$SCHEDULED TO 11
EQU PO$QTY TO 12
EQU PO$BALANCE TO 13
EQU PO$UNIT TO 14
EQU PO$DESC TO 15
EQU PO$VPN TO 16
EQU PO$ACCT TO 17
EQU PO$EST TO 18
EQU PO$NOTE TO 19
*
PRECISION 0 ;* For 5 digit prices
*
EQU vnum TO 2
EQU terms TO 3
EQU date TO 4
EQU part TO 5
EQU um TO 6
EQU est TO 7
EQU acct TO 8
EQU vpn TO 9
EQU desc TO 10
EQU note TO 11
EQU ddate TO 12
EQU qty TO 13
EQU price TO 14
EQU qa TO 15
EQU notes TO 16
EQU vname TO 17
EQU item TO 25
EQU dlv TO 26
EQU bal TO 27
EQU dtotl TO 28
EQU itotl TO 29
EQU ptotl TO 30
EQU iqty TO 31
EQU ok TO 32
EQU shopn TO 33
EQU shoqty TO 34
EQU shoprice TO 35
EQU shohdr TO 36
EQU shonotes TO 37
EQU shoum TO 38
EQU shoest TO 39
EQU shoacct TO 40
EQU shovpn TO 41
EQU shodesc TO 42
EQU shonote TO 43
EQU shodate TO 44
*
OPEN "DF" TO DF.FILE ELSE STOP "VAN1"
OPEN "PM" TO PM.FILE ELSE STOP "VAN2"
OPEN "PO" TO PO.FILE ELSE STOP "VAN3"
OPEN "VM" TO VM.FILE ELSE STOP "VAN4"
READ SCR FROM DF.FILE, "#GET.PO" ELSE STOP "VAN5"
*
AT.ERR = @(0,23):CHAR(27):"K"
ALREADY = AT.ERR:"Shipments already received!"
*
10 * Clear screen, start purchase order
RELEASE
BREAK KEY ON
PRINT CHAR(12):
*
20 * Get PO number
PRINT @(59,0) : TIMEDATE() :
SOMETHING.IN = 0 ;* For QA change test
ITEM = 1 ; DLV = 1 ; NOTE.VAL = 1
LOOP
  INPUT PO.ID USING SCR, "" ELSE STOP
  PO.ID = PO.ID<1>
  ITEM.LOCKED = 0
  READU PO FROM PO.FILE, PO.ID LOCKED ITEM.LOCKED = 1 ELSE PO = ""
WHILE ITEM.LOCKED DO
  PRINT AT.ERR:"PO group locked!":
REPEAT
NEW.PO = (PO="")
IF NOT(NEW.PO) THEN
  I = 1
  LOOP UNTIL PO<PO$ITEM,I> = "" DO
    IF PO<PO$RECEIVED,I,1> # "" THEN SOMETHING.IN = 1
    I = I+1
  REPEAT
  VM.ID = PO<PO$VM.ID> ; GOSUB 260 ;* Get vendor record
  IF VM.ERR THEN RELEASE ; GO TO 20
  INPUT IGNORE USING SCR, PO AT shohdr ELSE NULL
  INPUT IGNORE USING SCR, PO<PO$NOTES,1> AT shonotes ELSE NULL
  GOSUB 250 ; GOSUB 270 ;* Display item fields
  GOSUB 300 ;* Display delivery fields
  GO TO 200 ;* Show notes and get command
  END
*
INPUT IGNORE USING SCR, ITEM AT item ELSE NULL
INPUT IGNORE USING SCR, DLV AT dlv ELSE NULL
PO.PRICE = 0
*
30 * Get vendor number
```

```
LOOP
  INPUT VM.ID USING SCR, "" AT vnum ELSE GO TO 10
  VM.ID = VM.ID<1>
  GOSUB 260 ;* Get vendor master
WHILE VM.ERR DO REPEAT
PO<PO$VM.ID> = VM.ID
*
40 INPUT PO USING SCR, PO AT terms ELSE GO TO 210
50 INPUT PO USING SCR, PO AT date ELSE GO TO 210
*
60 * Get part
OLD.PART = PO<PO$PART,ITEM>
INPUT PART.NUM USING SCR, OLD.PART AT part ELSE GO TO 170
PART.NUM = PART.NUM<1>
PO<PO$ITEM,ITEM> = ITEM
PO<PO$PART,ITEM> = PART.NUM
IF PART.NUM # "" THEN
  READ PM FROM PM.FILE, PART.NUM ELSE
    PRINT AT.ERR:"Part ":PART.NUM:" not found!":
    * For now, allow input of non-existant parts
    PM = ""
    END
  I = 1
  LOOP UNTIL PO<PO$ITEM,I> = "" DO
    IF (PO<PO$PART,I>=PART.NUM) AND (I#ITEM) THEN ;* Warn buyer
      PRINT AT.ERR:"Part also on item #":I:"!":
      END
    I = I+1
  REPEAT
  IF OLD.PART = "" THEN
    PO<PO$DESC,ITEM> = PM<PM$DESC>
    PO<PO$UM,ITEM> = PM<PM$UM>
    GOSUB 280 ;* Show UM and desc
    END
  END
*
70 INPUT VALUE USING SCR, PO<PO$UM,ITEM> AT um ELSE GO TO 170
PO<PO$UM,ITEM> = VALUE<1>
80 INPUT VALUE USING SCR, PO<PO$EST,ITEM> AT est ELSE GO TO 170
PO<PO$EST,ITEM> = VALUE<1>
90 INPUT VALUE USING SCR, PO<PO$ACCT,ITEM> AT acct ELSE GO TO 170
PO<PO$ACCT,ITEM> = VALUE<1>
100 INPUT VALUE USING SCR, PO<PO$VPN,ITEM> AT vpn ELSE GO TO 170
PO<PO$VPN,ITEM> = VALUE<1>
110 INPUT VALUE USING SCR, PO<PO$DESC,ITEM> AT desc ELSE GO TO 170
PO<PO$DESC,ITEM> = VALUE<1>
120 INPUT VALUE USING SCR, PO<PO$NOTE,ITEM> AT note ELSE GO TO 170
PO<PO$NOTE,ITEM> = VALUE<1>
*
130 * Get delivery
OLD.DATE = PO<PO$SCHEDULED,ITEM,DLV>
LOOP
  INPUT DLV.DATE USING SCR, OLD.DATE AT ddate ELSE GO TO 160
  DLV.DATE = DLV.DATE<1>
  IF (DLV.DATE="") AND (OLD.DATE="") THEN GO TO 160 ;* Exiting
  IF DLV.DATE = OLD.DATE THEN GO TO 140 ;* No change
  IF (DLV.DATE="") AND (OLD.DATE#"") THEN ;* Deleting
    GOSUB 220 ;* Check receipts
    IF RECEIVED THEN GOSUB 300 ; GO TO 210 ;* Can't drop, refresh
    GOSUB 240 ;* Delete delivery
    GOSUB 250 ;* Item and PO total
    GOSUB 300 ;* Display delivery
    GO TO 130 ;* Get another date
    END
  * New date, with or without old date
  SAME.DATE = 1
  LOCATE DLV.DATE IN PO<PO$SCHEDULED,ITEM>,1 BY "AR"  SETTING NEW.
POS ELSE SAME.DATE = 0
  IF SAME.DATE THEN PRINT AT.ERR:"Delivery already scheduled for "
:OCONV(DLV.DATE,"D2-"):"!": ELSE
    IF PO<PO$QTY,ITEM,NEW.POS> # PO<PO$BALANCE,ITEM,NEW.POS> THEN
      * New date falls between receipts
      PRINT ALREADY:
      SAME.DATE = 1 ;* Treat as a collision
      END
    END
WHILE SAME.DATE DO REPEAT
INS DLV.DATE BEFORE PO<PO$SCHEDULED,ITEM,NEW.POS>
INS PO<PO$QTY,ITEM,DLV> BEFORE PO<PO$QTY,ITEM,NEW.POS>
INS PO<PO$BALANCE,ITEM,DLV> BEFORE PO<PO$BALANCE,ITEM,NEW.POS>
INS PO<PO$UNIT,ITEM,DLV> BEFORE PO<PO$UNIT,ITEM,NEW.POS>
IF NEW.POS <= DLV THEN DLV = DLV+1 ;* Old date just moved
IF OLD.DATE # "" THEN GOSUB 240 ;* Delete old delivery
IF NEW.POS > DLV THEN NEW.POS = NEW.POS-1
DLV = NEW.POS
GOSUB 250 ;* Item and PO total
GOSUB 300 ;* Show delivery
*
140 * Get qty
OLD.QTY = PO<PO$QTY,ITEM,DLV>
LOOP
  INPUT QTY USING SCR, OLD.QTY AT qty ELSE GO TO 160
  QTY = QTY<1>
WHILE QTY <= 0 DO
  PRINT AT.ERR:"Delete date to delete delivery!":
REPEAT
OLD.BAL = PO<PO$BALANCE,ITEM,DLV>
NEW.BAL = OLD.BAL + (QTY - OLD.QTY)
GOSUB 220 ;* Check for receipts, may warn when really OK
IF (NEW.BAL<0) OR (RECEIVED AND (OLD.BAL=0)) THEN GO TO 140
* Can't lower qty to cause negative balance. OK to change qty if
* partially received, but not if fully received (since shipment
* is then complete).
* Even if OK, error message might still be up as a warning.
PO<PO$BALANCE,ITEM,DLV> = NEW.BAL
PO<PO$QTY,ITEM,DLV> = QTY
GOSUB 310 ;* Delivery balance and total
GOSUB 250 ;* Item and PO total
*
150 * Get price
```

**GET.PO** LISTING CONTINUED

```
OLD.PRICE = PO<PO$UNIT,ITEM,DLV>
IF (OLD.PRICE="") AND (DLV>1) THEN
  OLD.PRICE = PO<PO$UNIT,ITEM,DLV-1>
  INPUT IGNORE USING SCR, (PO<PO$QTY,ITEM,DLV>*OLD.PRICE) AT dtotl
 ELSE NULL
  END
INPUT PRICE USING SCR, OLD.PRICE AT price ELSE GO TO 160
PO<PO$UNIT,ITEM,DLV> = PRICE<1>
GOSUB 250 ;* Total PO and item
DLV = DLV+1
GOSUB 300 ;* Display delivery fields
GO TO 130
*
160 * Exited delivery loop
* Delete delivery if unfinished
IF (PO<PO$QTY,ITEM,DLV>="") ! (PO<PO$UNIT,ITEM,DLV>="") THEN GOSUB
  240
ITEM = ITEM+1 ; GOSUB 250 ; GOSUB 270 ;* Show next item
DLV = 1 ; GOSUB 300 ;* Show delivery
GOSUB 230 ;* Check receipts
IF RECEIVED THEN GO TO 210 ;* Can't change item, receipts present
* No receipts, edits ok
GO TO 60
*
170 * Exited part loop
IF (PO<PO$UM,ITEM>="") ! (PO<PO$EST,ITEM>="") ! (PO<PO$ACCT,ITEM>=
"") ! (PO<PO$DESC,ITEM>="") THEN
  * At least one required field empty, item not finished
  GOSUB 290 ;* Delete item
  END
IF PO<PO$ITEM,1>="" THEN GO TO 10 ;* Must have at least one item
GOSUB 270 ; GOSUB 250 ;* Display item fields
GOSUB 300 ;* Display delivery fields
*
180 INPUT PO USING SCR, PO AT qa ELSE GO TO 210
190 *
LOOP
  INPUT VALUE USING SCR, PO<PO$NOTES,NOTE.VAL> AT notes ELSE VALUE
 = "EX"
  VALUE = VALUE<1>
UNTIL VALUE = "EX" DO
  PO<PO$NOTES,NOTE.VAL> = VALUE
  NOTE.VAL = NOTE.VAL+1
REPEAT
NOTE.VAL=1
200 INPUT IGNORE USING SCR, PO<PO$NOTES,NOTE.VAL> AT shonotes ELSE
  NULL
*
210 * Choose item and delivery
LOOP
  INPUT COMMAND USING SCR, "" AT ok ELSE GO TO 10
  COMMAND = COMMAND<1>
  LEFT.PART = OCONV(COMMAND,"G.1") ;* Item number or command
UNTIL LEFT.PART = "FI" DO
  BEGIN CASE
  CASE INDEX(COMMAND,".",1) # 0
    LEFT.PART = ICONV(LEFT.PART,"MD0")
    IF LEFT.PART < 1 THEN LEFT.PART = ITEM
    ITEM = 1
    LOOP UNTIL (PO<PO$ITEM,ITEM>="") OR (ITEM=LEFT.PART) DO
      ITEM = ITEM+1
    REPEAT
    RIGHT.PART=ICONV(OCONV(COMMAND,"G1.1"),"MD0")
    IF RIGHT.PART < 1 THEN RIGHT.PART = 1
    DLV = 1
    LOOP UNTIL (PO<PO$SCHEDULED,ITEM,DLV>="")!(DLV=RIGHT.PART) DO
      DLV=DLV+1
    REPEAT
    GOSUB 270 ;* Display item fields
    GOSUB 300 ;* Display delivery fields
  CASE LEFT.PART = "DI"
    DEL.ITEM = ITEM
    LOOP
      GOSUB 230 ;* Check receipts
    UNTIL RECEIVED OR (PO<PO$ITEM,ITEM>="") DO
      ITEM=ITEM+1
    REPEAT
    IF NOT(RECEIVED) THEN
      ITEM = DEL.ITEM
      GOSUB 290 ;* Delete item
      END
    GOSUB 270 ; GOSUB 250 ;* Display item and item total
  CASE LEFT.PART = 2
    IF SOMETHING.IN THEN PRINT ALREADY: ELSE
      RELEASE VM.FILE, VM.ID
      GO TO 30
      END
  CASE LEFT.PART = 3 ; GO TO 40
  CASE LEFT.PART = 6 ; GO TO 50
  CASE LEFT.PART = 12 ; GOSUB 230 ; IF NOT(RECEIVED) THEN GO TO 60
  CASE (13 <= LEFT.PART) AND (LEFT.PART <= 18)
    IF PO<PO$ITEM,ITEM> # "" THEN
      GOSUB 230 ;* Check receipts
      IF NOT(RECEIVED) THEN
        BEGIN CASE
        CASE LEFT.PART = 13 ; GO TO 70
        CASE LEFT.PART = 14 ; GO TO 80
        CASE LEFT.PART = 15 ; GO TO 90
        CASE LEFT.PART = 16 ; GO TO 100
        CASE LEFT.PART = 17 ; GO TO 110
        CASE LEFT.PART = 18 ; GO TO 120
        END CASE
        END
      END ELSE PRINT AT.ERR:"Part?":
  CASE LEFT.PART = 19 ; GOSUB 220 ; IF NOT(RECEIVED) THEN GOTO 130
  CASE (LEFT.PART=20) OR (LEFT.PART=21)
    IF PO<PO$SCHEDULED,ITEM,DLV> # "" THEN
      BEGIN CASE
      CASE LEFT.PART = 20 ; GO TO 140
      CASE LEFT.PART = 21 ; GOSUB 220 ; IF NOT(RECEIVED) THEN GOTO
      150
```

**GET.PO LISTING CONTINUED**

```
      END CASE
      END ELSE PRINT AT.ERR:"Delivery date?":
  CASE LEFT.PART = 27
    IF SOMETHING.IN THEN PRINT ALREADY: ELSE GO TO 180
  CASE LEFT.PART = 29 ; GO TO 190
  CASE LEFT.PART = ""
    IF PO<PO$SCHEDULED,ITEM,DLV> = "" THEN
      DLV=1
      IF PO<PO$ITEM,ITEM>="" THEN ITEM=1 ELSE ITEM=ITEM+1
      GOSUB 250 ; GOSUB 270 ;* Display item fields
      END ELSE DLV=DLV+1
    GOSUB 300 ;* Display delivery fields
    IF PO<PO$NOTES,NOTE.VAL> = "" THEN NOTE.VAL = 1 ELSE NOTE.VAL
= NOTE.VAL+1
    GO TO 200 ;* Show notes and restart command loop
  CASE 1
    PRINT AT.ERR:"What?":
  END CASE
REPEAT
*
IF PO<PO$TERMS> = "" THEN GO TO 40
IF PO<PO$DATE> = "" THEN GO TO 50
IF PO<PO$ITEM,1> = "" THEN GO TO 60
IF PO<PO$QA> = "" THEN GO TO 180
BREAK KEY OFF
GOSUB 250 ;* Get PO total
PO<PO$TOTAL> = PO.PRICE
RELEASE VM.FILE, VM.ID
WRITE PO ON PO.FILE, PO.ID
GO 10
*
220 * Check if delivery already has receipts
RECEIVED = (PO<PO$QTY,ITEM,DLV> # PO<PO$BALANCE,ITEM,DLV>)
IF RECEIVED THEN PRINT ALREADY:
RETURN
*
230 * Check if item already has receipts
RECEIVED = (PO<PO$RECEIVED,ITEM,1> # "")
IF RECEIVED THEN PRINT ALREADY:
RETURN
*
240 * Delete delivery data
DEL PO<PO$SCHEDULED,ITEM,DLV>
DEL PO<PO$QTY,ITEM,DLV>
DEL PO<PO$BALANCE,ITEM,DLV>
DEL PO<PO$UNIT,ITEM,DLV>
RETURN
*
250 * Get and show PO total
ITEM.QTY = 0 ; ITEM.PRICE = 0 ; PO.PRICE = 0
I = 1
LOOP UNTIL PO<PO$ITEM,I> = "" DO
  D = 1
  LOOP WHILE PO<PO$SCHEDULED,I,D> # "" DO
    Q = PO<PO$QTY,I,D>
    * Unit cost is to 5 decimals, so OCONV call makes_it pennies
    P = OCONV(Q * PO<PO$UNIT,I,D>,"MD03")
    IF I = ITEM THEN
      ITEM.QTY = ITEM.QTY + Q
      ITEM.PRICE = ITEM.PRICE + P
      END
    PO.PRICE = PO.PRICE + P
    D = D+1
  REPEAT
  I = I+1
REPEAT
INPUT IGNORE USING SCR, ITEM.QTY AT iqty ELSE NULL
INPUT IGNORE USING SCR, ITEM.PRICE AT itotl ELSE NULL
INPUT IGNORE USING SCR, PO.PRICE AT ptotl ELSE NULL
RETURN
*
260 * Get vendor master
VM.ERR = 1
READU VM FROM VM.FILE, VM.ID LOCKED
  PRINT AT.ERR:"Vendor ":VM.ID:" in use, try later!":
  RETURN
  END ELSE
    PRINT AT.ERR:"Vendor ":VM.ID:" not found!":
    RETURN
    END
VM.ERR = 0
INPUT IGNORE USING SCR, VM AT vname ELSE NULL
RETURN
*
270 * Display item
INPUT IGNORE USING SCR, ITEM AT item ELSE NULL
INPUT IGNORE USING SCR, PO<PO$PART,ITEM> AT shopn ELSE NULL
INPUT IGNORE USING SCR, PO<PO$EST,ITEM> AT shoest ELSE NULL
INPUT IGNORE USING SCR, PO<PO$ACCT,ITEM> AT shoacct ELSE NULL
INPUT IGNORE USING SCR, PO<PO$VPN,ITEM> AT shovpn ELSE NULL
INPUT IGNORE USING SCR, PO<PO$NOTE,ITEM> AT shonote ELSE NULL
280 * Enter here for UM and desc
INPUT IGNORE USING SCR, PO<PO$UM,ITEM> AT shoum ELSE NULL
INPUT IGNORE USING SCR, PO<PO$DESC,ITEM> AT shodesc ELSE NULL
RETURN
*
290 * Delete item
DEL PO<PO$ITEM,ITEM>
DEL PO<PO$PART,ITEM>
DEL PO<PO$UM,ITEM>
DEL PO<PO$EST,ITEM>
DEL PO<PO$ACCT,ITEM>
DEL PO<PO$VPN,ITEM>
DEL PO<PO$DESC,ITEM>
DEL PO<PO$NOTE,ITEM>
RETURN
*
300 * Display delivery
INPUT IGNORE USING SCR, DLV AT dlv ELSE NULL
```

**GET.PO LISTING CONTINUED**

## GET.PO LISTING CONTINUED

```
431 INPUT IGNORE USING SCR, PO<PO$SCHEDULED,ITEM,DLV> AT shodate ELSE
     NULL
432 INPUT IGNORE USING SCR, PO<PO$QTY,ITEM,DLV> AT shoqty ELSE NULL
433 INPUT IGNORE USING SCR, PO<PO$UNIT,ITEM,DLV> AT shoprice ELSE NULL
434 310 * Enter here for balance and total
435 INPUT IGNORE USING SCR, PO<PO$BALANCE,ITEM,DLV> AT bal ELSE NULL
436 INPUT IGNORE USING SCR, (PO<PO$QTY,ITEM,DLV>*PO<PO$UNIT,ITEM,DLV>)
     AT dtotl ELSE NULL
437 RETURN
438 *
439 END
```

Ⓟ

```
001
002
003 PO]VEN]TERMS]DATE]PN]UM]EST]ACCOUNT]VPN]DESC]NOTE]DDATE]QTY]UNIT]Q
    A]NOTES]VNAME]VADR1]VADR2]VCITY]VST]VZIP]VCONT]VPHONE]ITEM]DLV]BAL
    ]DTOTL]ITOTL]PTOTL]IQTY]OK]SHOPN]SHOQTY]SHOPRICE]SHOHDR]SHONOTES]S
    HOUM]SHOEST]SHOACCT]SHOVPN]SHODESC]SHONOTE]SHODATE
004
005 Create or change purchase orders\\1 PO #:              2 Ven #:
         Name:\                                      Adr1:\3 Terms:
                                Adr2:\
       City:                          St:    ZIP:\
                Cont:                              Ph:\\6 Date:\\Item:      12 P
    art:                13 UM:      14 Est:                15 Account:\16 VP
    N:                                 17 Desc:\
                  18 Note:\\Dlv:     19 Date:                 20 Qty:          Ba
    lance:         21 Unit price:\\Dlv Total:                  Item Qty:
     Item Total:                PO Total:\\\27 QA:        29 Notes:
006 0,0
007 ]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]PN]QTY]UNIT]VEN\TERMS\DATE\QA]NOTE
    S]UM]EST]ACCOUNT]VPN]DESC]NOTE]DDATE
008
009 S]S]S]S]S]S]S]S]S]S]S]S]S]S]S]S]S]S]S]S]S]S]S]S]S]S]S]S]S]S]S]N]
    N]N]N]N]N]N]N]N]N]N]N
010
011 L]L]L]L]L]L]L]L]L]L]L]L]L]L]L]L]L]L]L]L]L]L]L]L]L]L]L]L]L]L]L]L]L
012 1]1]4]2]1]1]1]1]1]1]1]1]1]5]1]2]3]4]5]6]7]9]8]1]1]1]1]1]1]1]1
013 Y]Y]Y]Y]]Y]Y]Y]]Y]]]Y]Y]Y
014
015
016 2,7]2,28]4,8]8,7]10,18]10,39]10,51]10,74]11,7]11,45]12,45]14,16]14
    ,34]14,71]19,6]19,21]2,42]3,42]4,42]5,42]5,67]5,74]6,42]6,67]10,5]
    14,4]14,50]16,10]16,49]16,69]16,30]22,68
017 7]4]15]9]7]2]9]5]26]30]35]9]6]8]1]58]37]37]38]21]2]5]21]12]2]2]6]1
    0]10]10]7]7
018
019 ]]]D2-]]]MD5]]]]]D2-]]MD5]]]]]]]]]]]]]]]MD25$]MD2$]MD2$
020 7]4]15]9]7]2]9]5]26]30]35]9]6]8]1]58]37]37]38]21]2]5]21]12]2]2]6]1
    0]10]10]7]7
021 ]]]]]]]]]]]]]]]]2]3]4]5]6]7]9]8]1]1]1]1]1]1]1
022 1 PO #:]2 Ven #:]3 Terms:]6 Date:]12 Part:]13 UM:]14 Est:]15 Accou
    nt:]16 VPN:]17 Desc:]18 Note:]19 Date:]20 Qty:]21 Unit price:]27 Q
    A:]29 Notes:]]]]]]]]]]]]]]]]File (FI) exit (EX) drop itm (DI) chg
    (#) goto (I.D) cycle (RETURN):
023 22,0]22,0]22,0]22,0]22,0]22,0]22,0]22,0]22,0]22,0]22,0]22,0]22,0]2
    2,0]22,0]22,0]]]]]]]]]]]]]]]]]22,0
024 22,7]22,8]22,8]22,7]22,8]22,6]22,7]22,11]22,7]22,8]22,8]22,8]22,7]
    22,14]22,6]22,9]]]]]]]]]]]]]]]]]22,68
025 73]72]72]73]72]74]73]69]73]72]72]72]73]66]74]71]]]]]]]]]]]]]]]]]12
026
027 7]4]15]9]7]2]9]5]26]30]35]9]6]8]2]58]]]]]]]]]]]]]]]7
028 ]](1NON'%'1NON'N'1NON)0('N'1NON)0('NOTE')](D)]](2A)]](5N)]]]](D)]]
    ]('Y')0('N')]]]]]]]]]]]]]]]]]]('FI')0('DI')0(R2,29)0(0N'.'0N)
029
030
031 MD0]MD0]]D]]]MD5]]]]]D]MD0]MD5
032 F]F]F]F]F]F]F]F]F]F]F]F]F]F]]]]]]]F]F]F]F]F]F]F]F]F]F]F]F]F
    ]F]F]F]F]F]F
033
034
035
036
037
038
039
040 ('EX')E
```

## GET.PO SCREEN DEFINITIONS

Ⓟ

**FIXED ASSET/PREPAID EXPENSE ACCOUNTING MODULE**

By Diogenes Computer Systems, Inc.
Specialists — Properties Management
Real Estate Construction

* Easy to use
* Hardcopy Validation/Backup
* Flexible Accounting Periods
* Book, Tax and State Calculations
* Supports all IRS Approved Methods
* 2 Years of Proven Reliability
* Built in Security
* Complete User and Program Documentation
* Stand-Alone or Integrate with your G/L
* Includes 1981 Economic Recovery Tax Act
* Recapture Calculated Automatically
* Depreciation Forecasting Module
* Demonstrations Available in New York Metropolitan Area

**AFFORDABLY PRICED AT $2500.00**

Integration and Implementation Available

For More Information —
Call — (201) 764-7282 8 A.M.—8 P.M.
Write — Diogenes Computer Systems, Inc.
P.O. Box 667
Highland Lakes, N.J. 07422

Software Licensing Inquiries Invited

**WIZARD**

**Introducing Two Important New Products:**
**WIZARD/REPORTS and WIZARD/MENUS**

The WIZARD screen generator's superiority has been established by the fact that it has been chosen by ADDS and Intertechnique to be a standard feature on their PICK systems. Several other manufacturers are making similar plans.

Now API introduces two new products:
WIZARD/REPORTS and WIZARD/MENUS.

WIZARD/REPORTS goes far beyond the restrictive columnar approach of the PICK report generator by allowing information to be displayed anywhere on the report. The WIZARD/REPORTS formatter pages horizontally and vertically, allowing you to display the entire width and length of your report layout on an 80 column screen as you are developing it.

WIZARD/MENUS quickly generates easy to use menus with optional security features.

For a 30 day trial of the entire family of WIZARD products for $100, call AUTOMATIC PROGRAMMING INC. at 714-552-2800.

**Do you know what your computer is REALLY doing? You can with SAM! SAM knows and tells, in realtime!**

The **S**ystem **A**ctivity **M**onitor from Burns & Associates, displays disk and CPU activity data for ALL tasks active on your Microdata system. **SAM** lets you see what tasks are slowing your system down and gives you the information you need to know to assess the need for changes and upgrades.

**SAM** has been in use for over a year at various sites and is now available for your 3.X or 4.X Microdata system for $350.00. Shipment within 5 days of receipt of check, tape density, operating system level and serial number. Multiple system discounts available, please call.

Systems houses: We have a good source code encryption program available to protect your software investment.

**Burns & Associates**
Box 451
Cincinnati, Ohio 45201
**606/341-0640**

**• THE SIDEKICK •**

**A Microcomputer that talks to your Microdata**

**Now a desktop microcomputer can communicate with your Microdata REALITY via the standard PRISM port.**

**USE THE FILE TRANSFER MODE TO:**

- PASS INFORMATION FROM MICRODATA FILES TO A FULL-FLEDGED WORK PROCESSOR SUCH AS WORDSTAR
- SUPPORT REMOTE DATA ENTRY WITH FULL DATA VALIDATING AND SCREEN FORMATTING LOGIC. DIAL UP THE HOST AND TRANSMIT FILES PERIODICALLY.
- PERFORM A VARIETY OF TELECOMMUNICATIONS FUNCTIONS BY TRANSFERRING FILES TO/FROM THE MICRODATA AND USING THE PROGRAMMABLE PORTS ON THE MICROCOMPUTER.

**USE THE INTERACTIVE MODE TO:**

- LIGHTEN THE LOAD ON YOUR MICRODATA BY DISTRIBUTING PROCESSING WHILE STILL PERMITTING INQUIRY AND UPDATING OF MICRODATA FILES.
- PROVIDE MULTIPLE FUNCTIONS ON ONE PRISM® PORT. ONE MICROCOMPUTER CAN ACT AS A PRISM®, LOG TRANSACTIONS TO A FLOPPY DISK, AND DRIVE A PRINTER.

Programs are presently available for Televideo and Intertec Superbrain® . Others will be developed on request.

**TURNKEY HARDWARE/SOFTWARE PACKAGES AVAILABLE.**

Ben Ansbacher
AnsData Associates
P.O. Box 2598
Burlington, N.C. 27216-2598
**800-334-6823 or 919-227-0123**

PRISM and REALITY are trademarks of Microdata Corp.
WordStar is a trademark of MicroPro International Corp.
Superbrain is a trademark of Intertec Data Systems Corp.

## games

**For many people, computers are synonymous with games, now that the video game industry has become such a giant. Programmers frequently cut their teeth on small game programs, since such programs are often straight-forward and self-contained without a lot of complicated interfaces to files or other software. And, more than anything, games are simply entertaining and fun.**

**The Games Department will be making periodic appearances in Pragma. If you have a game program you would like to share, send it in for publication. If there's anything the Pragma staff has time to do, it's "performing rigorous software quality assurance" (in other words, trying out a new game on the computer).**

## Swat!

**The French mathematician and philosopher Rene Descartes generally stayed and worked in bed until midday for most of his life, because of poor health. Legend has it that Descartes invented the concept of co-ordinate geometry while resting in bed and watching a fly crawl across the ceiling. What mathematical notions do you think *you* can devise while watching a fly buzz across your display screen?**

Swat is both a game of skill and a game of chance. When the Swat program is executed, it displays a "user" (shown as an asterisk) and a fly (shown as a percent sign, since that looks kind of like an insect). The object of the game is to use the keyboard to move the user to the same position as the fly, thereby scoring a swat. Skill is required because the keyboard controls the user's *velocity* on the screen, not just position. Plus, the user can move off the edge of the screen out of the view of the display window. (Not much skill is required to move out of view, but it can be tricky to try and get back onto the display.) Chance is involved because each time the user moves, it generally scares the fly, which may randomly buzz to a new position. Since the fly trys to annoy the operator as much as possible, it always remains on the screen and avoids flying out of view.

To move the asterisk on the screen, the user must hit one of the numeric keys labeled from 1 to 9. It's helpful to use a three by three numeric pad, since then the position of the chosen key corresponds to the asterisk's change in direction and acceleration: hitting 1 means apply more acceleration toward the upper left, 2 means accelerate straight up, 3 means accelerate toward the upper right, and so on around the keypad. Hitting the 5 key means don't accelerate, just keep moving with the current velocity.

To help the user get the hang of moving about the screen, the program optionally offers two special modes of operation: cheat mode and trace mode. In cheat mode, the program displays the digits 1 to 9 on the screen, indicating where the asterisk will end up once the user hits one of the indicated keys. Cheat mode can be turned on and off by hitting the C key. Trace mode, which is toggled on and off with the T key, causes the asterisk to leave behind a period on the screen each time a move is made. The periods trace out the path the asterisk has taken. For a graphic lesson in how to control your motion in Swat, start the program running and try to ignore the distracting fly for a moment. Hit the T key to turn on tracing mode, then hit the 6 key to start moving to the right. Now carefully follow this sequence of keystrokes: hit the 8 key three times, hit the 2 key six times, hit 8 six times, hit 2 six times, 8 six times, 2 six times, 8 six times, and so on. Take your time entering the numbers, making sure you give the system plenty of time to update the screen and prompt for your next input, especially if your terminal is running any slower than 9600 baud. If you correctly follow the sequence of keystrokes, you'll see that the periods soon trace out a smooth waveform on the screen. Always remember that a period (in trace mode) shows where you were, the asterisk shows where you are, and a digit (in cheat mode) shows where you'll be.

The listing for Swat shown on page 46 is set up for a 24 by 80 character screen, but can easily be adjusted for any size display by changing the numbers in line 2. Line 3 steals one line from the bottom of the display for use in prompting. The XUSER, YUSER, and XFLY, XFLY variables are initialized in lines 6 and 7, and represent the coordinates of the user and the fly. XNEW, YNEW are set in line 10 to predict the user's new position if the user's current X and Y components of velocity (XVEL and YVEL, which are initialized in line 8) are added to the current position.

If cheat mode is on, then subroutine 200 is called in line 11 to display the keypad choices on the screen. The DIGITS variable is set to 1 before calling routine 200 to make line 66 paint each digit, otherwise DIGITS can be set to zero to make line 66 erase the digits displayed by a previous call, such as at line 27. Line 64 makes sure a PRINT is attempted only if each digit is actually in the window. Similarly, the VISIBLE variable initialized in line 13 makes sure the asterisk is only plotted at line 15 if the user's position is on the screen.

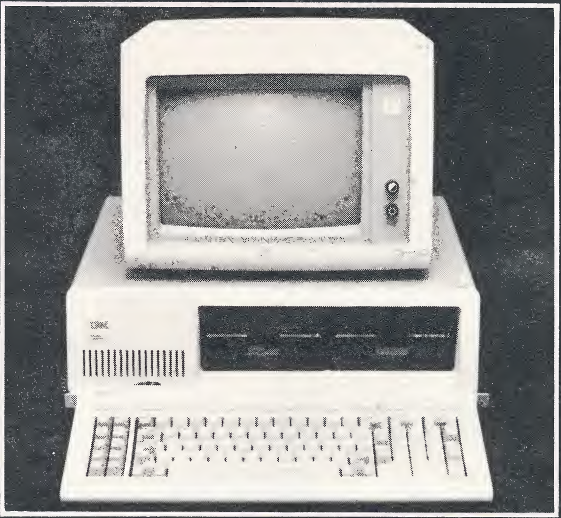
The game ends at line 17 if the user and fly finally rest at the same position, as tested in line 16. Otherwise, the prompt line is built starting at line 20. The STEPS variable bumped in line 26 tracks the number of moves made by the user. Any kind of input is counted as a move. The CASE statement starting at line 28 adjusts the user's velocity if 1 to 9 (but not 5) is input, toggles cheat and trace modes if C or T is input, quits on Q, and resets on R (handy if the user is hopelessly lost off the edge of the screen).

Line 45 erases the user's old position, or deposits a period for the same effect if tracing. Line 47 lets the fly hop from 0 to 19 times. Each hop, handled by lines 48 to 56, can be one or none squares in any direction, with the IF tests making sure the fly stays on the screen. Finally, line 57 is where the user's velocity components are actually added in, allowing the complete cheat-input-move loop to start again at label 100.

Swat is a simple enough game so that enhancements, such as obstacles on the screen, or real-time movement (such as supported by the special INPUT statement on Prime Information) while the user is deciding what to do, should be easy to add to the program. In the mean time, perhaps playing Swat as much as possible will help justify the acquisition of a joystick for your terminal.

P

## IBM's Personal Computer Talks to Your Minicomputer Via R/NET



You can connect an IBM PC equipped with R/NET to your Reality, Ultimate, Mentor, Evolution, or Information relational data base machine. R/NET executes your existing software without modification. In addition to supporting cursor control for your interactive programs, you can either print the current screen contents or engage the printer for printing a multi-page report. And under operator or program control, R/NET will capture data sent to the IBM PC on its own disk storage.

Now you can let VisiCalc do your financial modeling. Execute EasyWriter on your dedicated microcomputer for your wordprocessing requirements. Your PC is multi-lingual, i.e., BASIC, Cobol, Pascal, Forth, FORTRAN, C, and the 8086 Macro Assembler. Application packages like General Ledger, Accounts Receivable, Accounts Payable, Inventory Control, Job Costing, and Payroll are available today.

R/NET is distributed on its own diskette and comes complete with documentation manual and interface cable for $200. A typical PC configuration retails for $3175. Make your next terminal an IBM Personal Computer, the most respected name in the industry.

COSMOS

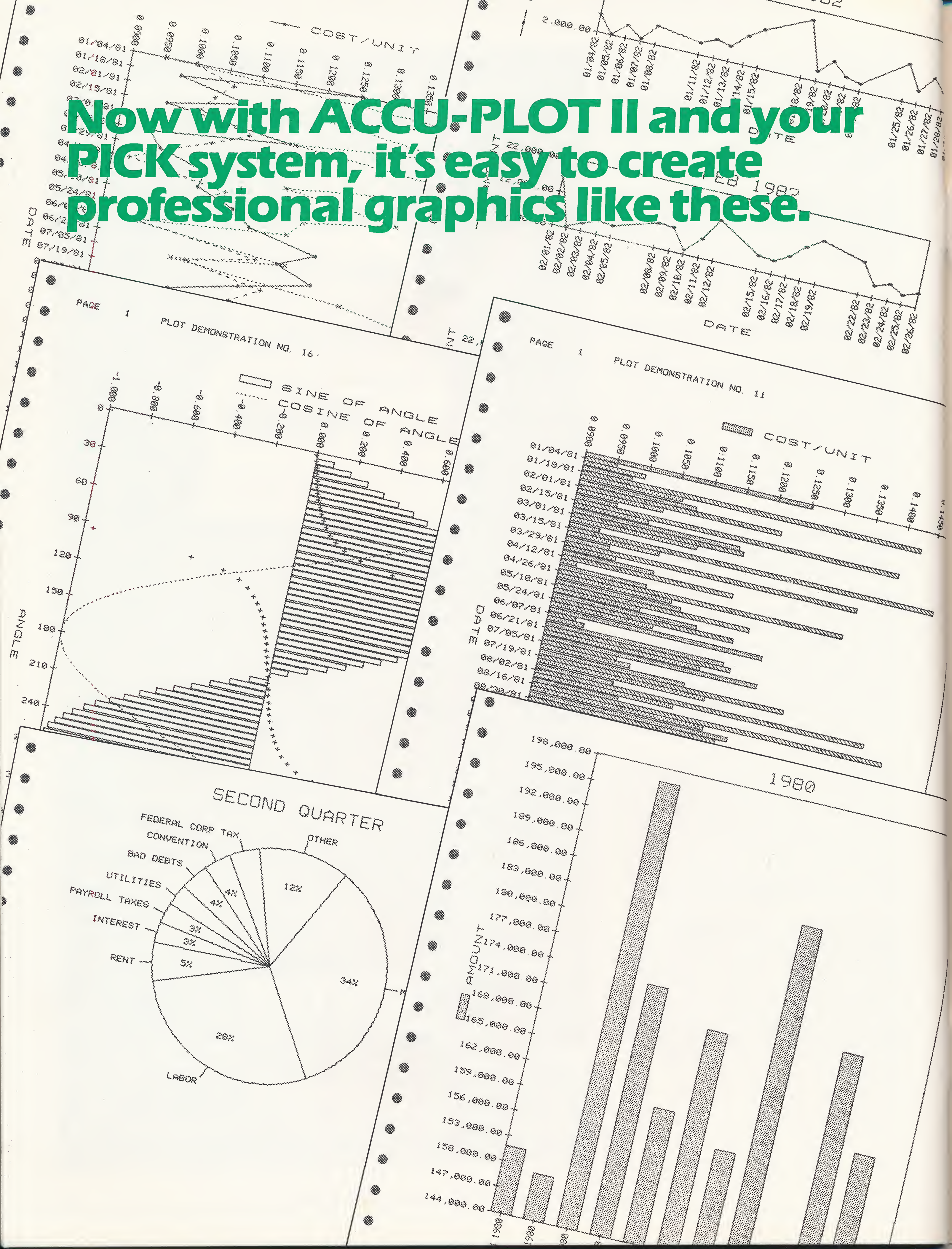
Direct your orders to: Cosmos Inc.,

Box AH, 123 Ferntree Dr. West, Morton, WA 98356
(206) 496-5974 • (206) 226-9362 for 24 hour answering.

**Think Relational**

• VisiCalc TM of VisiCorp • EasyWriter TM of Information Unlimited Software Inc. • R/NET TM of Cosmos Inc. • IBM TM of International Business Machines • Ultimate TM of Ultimate Corp. • Mentor TM of Applied Digital Data Systems • Evolution TM of Evolution Corp. • Information TM of Prime • Reality TM of Microdata Corp.

**Now with ACCU-PLOT II and your PICK system, it's easy to create professional graphics like these.**



## When it comes to communicating complex business information, an ACCU-PLOT picture is worth 1,000 words.

ACCU-PLOT II from Accu/Soft Enterprises is a new, easy to use graphics system that will produce bar charts, point-to-point line charts, scatter graph diagrams and pie charts in either black and white or color.

ACCU-PLOT II is as easy to use as ENGLISH. The sentences used to produce graphs are similar to the "LIST" and "SORT" verbs you use every day. No modification to your dictionaries or data files is required.

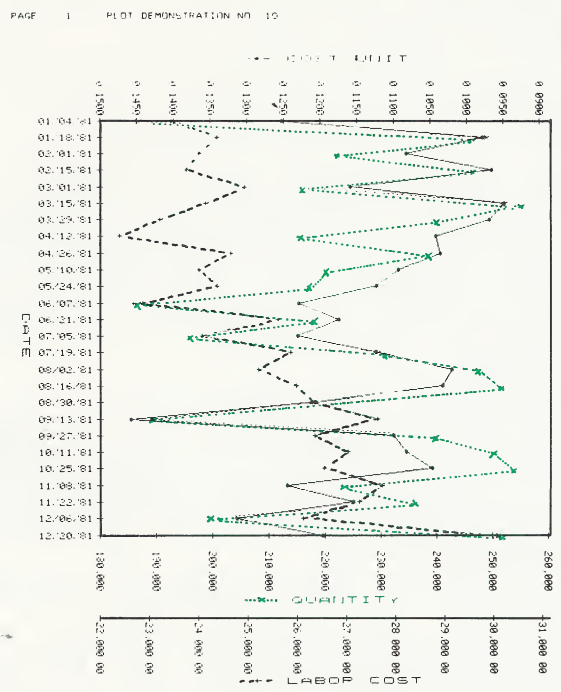
ACCU-PLOT II is available for Microdata, Ultimate, ADDS Mentor, Evolution, CDI's IBM Series/I, Datamedia, General Automation and other PICK based computer systems. ACCU-PLOT II will operate on most dot matrix printers with graphics capabilities as well as most graphic CRT's, including the IDS Prism printer and other devices capable of producing color graphics.

### Some of the key features of ACCU-PLOT II are:

- Syntax similar to the standard inquiry language formats (ENGLISH, RECALL, INFO/ACCESS, etc.)
- Standard data files and dictionaries.
- Any number of attributes can be plotted.
- Line, bar, or scatter formats may be mixed.
- Pie charts can show "exploded" slices.
- Automatic scaling with optional override.
- User-specified captions above and below graph.
- Horizontal and vertical formats are available.
- Multiple graphs can be placed on a single page.
- A single graph may be spread over multiple pages.
- Text data may be placed on the x-axis.
- Y-axis attributes may be grouped for common scaling.
- User definable line and bar styles and chart format.
- Four character sizes for dot matrix printers.
- Interfaces with BASIC programs for special applications.
- Interfaces with COMPU-SHEET electronic spreadsheet.

**ACCU/SOFT ENTERPRISES**
8655 Belford Avenue, Suite 100
Los Angeles, California 90045

### Here's how easy it is to create this chart:



**Simply enter:** SPLOT PLOTDEMO2 BY DATE ID-SUPP DATE COST/UNIT QUANTITY LABOR-COST LPTR HEADING "PAGE" 'P' PLOT DEMONSTRATION NO. 10 'LL' "

### Put the graphic power of ACCU-PLOT at your command. For more information, mail this coupon or call (213) 649-5800 today.

**Yes! Please send me more information about ACCU-PLOT II.**

Name ______________________________
Title ______________________________
Company ______________________________
Address ______________________________
City ______________________________
State ________________ Zip __________
Telephone (        ) ______________________

```
EQU clear TO CHAR(12)
XMAX = 79 ; YMAX = 23 ;* CRT maximums
YMAX = YMAX-1 ;* Leave room for control line
10 * Start new game
PRINT clear: ; CHEATING = 0 ; TRACING = 0 ; STEPS = 0
XUSER = INT(XMAX/5) ; YUSER = INT(YMAX/2)
XFLY = INT(XMAX/2) ; YFLY = YUSER
XVEL = 0 ; YVEL = 0
100 * Loop back to here for each move
XNEW = XUSER+XVEL ; YNEW = YUSER+YVEL
IF CHEATING THEN DIGITS = 1 ; GOSUB 200 ;* Show where choices lead
PRINT @(XFLY,YFLY):"%": ;* Show fly
VISIBLE = (0<=XUSER) & (XUSER<=XMAX) & (0<=YUSER) & (YUSER<=YMAX)
IF VISIBLE THEN ;* Human on screen, may have reached fly
   PRINT @(XUSER,YUSER):"*": ;* Show you
   IF (XUSER=XFLY) & (YUSER=YFLY) THEN
      PRINT clear:"SWAT! (":STEPS:" steps.)" ; STOP
      END
   END
PRINT @(0,YMAX+1):"You're @(":XUSER:",":YUSER:"), ":
IF CHEATING THEN PRINT "cheating, ":
IF TRACING THEN PRINT "tracing, ":
PRINT "fly's @(":XFLY:",":YFLY:"). ":
PRINT "Your move? (1-9,C,Q,R,T)":CHAR(27):"K": ;* Clear line
INPUT DIR,1: ;* Get direction to move
STEPS = STEPS+1
IF CHEATING THEN DIGITS = 0 ; GOSUB 200 ;* Erase hints
BEGIN CASE ;* Acceleration
CASE DIR=1 ; XVEL=XVEL-1 ; YVEL=YVEL+1
CASE DIR=2 ;                YVEL=YVEL+1
CASE DIR=3 ; XVEL=XVEL+1 ; YVEL=YVEL+1
CASE DIR=4 ; XVEL=XVEL-1
CASE DIR=5
CASE DIR=6 ; XVEL=XVEL+1
CASE DIR=7 ; XVEL=XVEL-1 ; YVEL=YVEL-1
CASE DIR=8 ;                YVEL=YVEL-1
CASE DIR=9 ; XVEL=XVEL+1 ; YVEL=YVEL-1
CASE DIR="C" ; CHEATING = NOT(CHEATING)
CASE DIR="Q" ; PRINT clear:STEPS:" steps for nothing." ; STOP
CASE DIR="R" ; GO TO 10
CASE DIR="T" ; TRACING = NOT(TRACING)
END CASE
IF VISIBLE THEN ;* Last position was on screen
   PRINT @(XUSER,YUSER):
   IF TRACING THEN PRINT ".": ELSE PRINT " ":
   END
HOPS = RND(20)
FOR I = 1 TO HOPS
   PRINT @(XFLY,YFLY):" ":
   XFLY = XFLY+RND(3)-1 ; YFLY = YFLY+RND(3)-1
   IF XFLY < 0 THEN XFLY = 0
   IF YFLY < 0 THEN YFLY = 0
   IF XFLY > XMAX THEN XFLY = XMAX
   IF YFLY > YMAX THEN YFLY = YMAX
   PRINT @(XFLY,YFLY):"%":
NEXT I
XUSER = XUSER+XVEL ; YUSER = YUSER+YVEL
GO TO 100
*
200 * Cheating
FOR J = -1 TO 1
   FOR I = -1 TO 1
      XCRT = XNEW+I ; YCRT = YNEW+J
      IF (0<=XCRT) & (XCRT<=XMAX) & (0<=YCRT) & (YCRT<=YMAX) THEN
         PRINT @(XCRT,YCRT):
         IF DIGITS THEN PRINT I+5-(3*J): ELSE PRINT " ":
         END
   NEXT I
NEXT J
RETURN
*
END
```

## SWAT PROGRAM LISTING

Ⓟ

AuroPlan T.M.

AUROTECH
OF COLORADO

## Now Software Similar To VisiCalc® Is Available On The Pick Operating System

**A financial planning and modeling system that surpasses all others. Ideal for preparing financial models, budgets, cost estimates, sales recaps, scheduling and resource reports and analyses.**

**AUROPLAN** is an electronic worksheet designed to drastically reduce the time and effort required to prepare financial reports.

**AUROPLAN** has a trial value feature which allows you to ask "What if?" without destroying previously entered data or formulas.

**AUROPLAN** allows you to name elements in one worksheet and reference them in another, so you can construct modular worksheets and link them together. This eliminates the need for huge worksheets and the long calculation times associated with them.

**AUROPLAN** can retrieve data directly from your existing computer data base without the need for costly interface programming.

**AUROPLAN** can be learned in just a few hours. The only limit to its capability is your imagination.

That's **AUROPLAN!** Now available for use with the Pick Operating System.

For more information on **AUROPLAN,** contact:

AUROTECH
OF COLORADO

925 S. Niagara
Denver, CO 80224
(303) 388-1612

VisiCalc® is a trademark of VisiCorp, Inc.

PRAGMA
207 GRANADA DRIVE
APTOS, CA 95003

BULK RATE
U.S. POSTAGE
PAID
APTOS, CA 95003
Permit No. 67

ADDRESS CORRECTION REQUESTED

INTRODUCING

HAL™

THE COMPREHENSIVE APPLICATION SYSTEM DEVELOPER

While most tools and "generators" available do only part of the job, HAL creates **entire** software applications. Note a few of HAL's features:

- Generates, with ease, menus, files, forms, reports, **all** driver logic (both arithmetic and control), and documentation manuals (technical documentaion is 100% automatic).
- Enables elaborate user/terminal/program security for every application.
- Provides a simple way to generate error messages, help messages, prompts, etc. throughout an application.
- Automatically organizes all elements of an application system for ease of reference and maintenance.
- Includes a powerful stack processor for complex operations.
- Provides built-in complete application design portability.
- Eliminates the need for a large capital expenditure . . . HAL is **rented** instead of purchased!

In short, HAL is the most advanced, user-friendly means ever available to create quality software for Pick-type systems. A demonstration can be arranged.

CHRONON DATA CORPORATION
Post Office Box 2225 ● Houston, Texas 77001 ● (713) 984-2765